

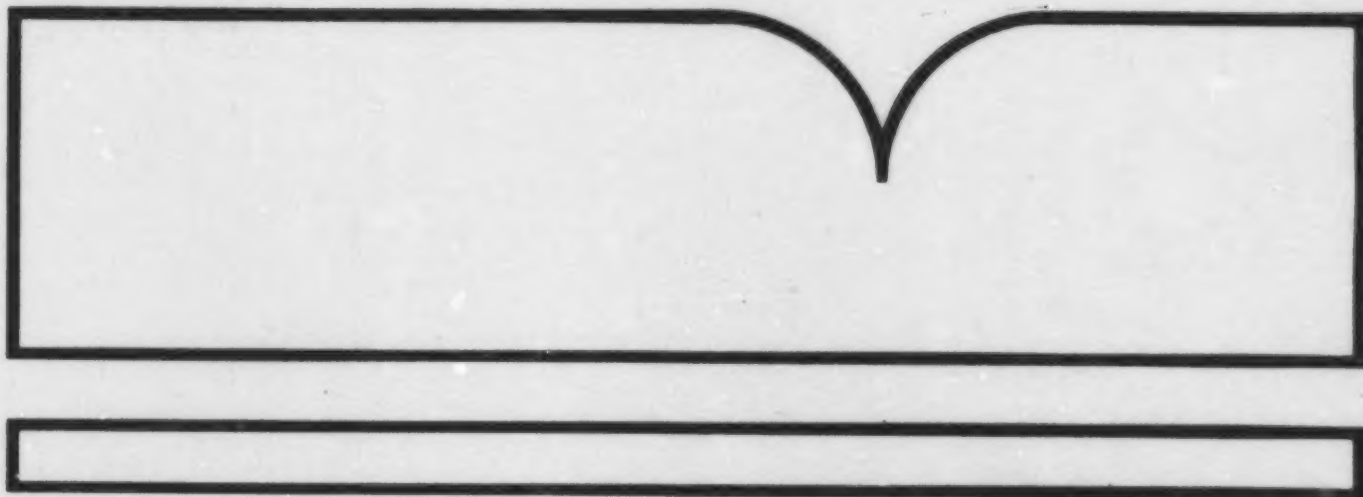
Economic Impact Analysis, RCRA
(Resource Conservation and Recovery Act)
Interim Status Standards. Volume IV

Arthur D. Little, Inc.
Cambridge, MA

Prepared for

Environmental Protection Agency
Washington, DC

1981



U.S. Department of Commerce
National Technical Information Service

NTIS

ECONOMIC IMPACT ANALYSIS
RCRA INTERIM STATUS STANDARDS

Volume - IV

This report was prepared by Arthur D. Little, Inc. under contract number 68-01-4671 for the Office of Solid Waste and Emergency Response.

REPRODUCED BY
NATIONAL TECHNICAL
INFORMATION SERVICE
U.S. DEPARTMENT OF COMMERCE
SPRINGFIELD, VA 22161

U.S. ENVIRONMENTAL PROTECTION AGENCY
1981

BIBLIOGRAPHIC INFORMATION

PB82-183526

Economic Impact Analysis, RCRA (Resource Conservation and Recovery Act) Interim Status Standards. Volume IV.

1981

PERFORMER: Little (Arthur D.), Inc., Cambridge, MA.
Contract EPA-68-01-4671

SPONSOR: Environmental Protection Agency, Washington, DC.

See also Volume 3, PB82-183518. Also available in set of 4 reports PC E99, PB82-183484.

This appendix contains two sets of sample calculations. The first example treats an industry segment, phenolic resins, according to the methodology used by Arthur D. Little, Inc., for the industries for which other contractors did not provide basic data. The second example treats an industry, leather tanning, for which, the basic data were provided by another contractor. The two methods differed somewhat -- particularly in the definition of model plant subcategories and in the way the subcategories are aggregated.

KEYWORDS: *Economic impact, *Hazardous materials,
*Industries.

Available from the National Technical Information Service,
Springfield, Va. 22161

PRICE CODE: PC A10/MF A01

APPENDIX H
SAMPLE CALCULATIONS

TABLE OF CONTENTS

VOLUME 4

	Page
APPENDIX H - SAMPLE CALCULATIONS	
1. EXAMPLE OF COST CALCULATIONS - THE PHENOLIC RESIN INDUSTRY SEGMENT	H-1
2. A SECOND EXAMPLE - LEATHER TANNING	H-39
APPENDIX I - COMPUTER PROGRAM	
1. UNIT COST FORMULAS	I-1
2. COMPUTER PROGRAM	I-1

APPENDIX H

SAMPLE CALCULATIONS

This appendix contains two sets of sample calculations. The first example treats an industry segment, phenolic resins, according to the methodology used by Arthur D. Little, Inc., for the industries for which other contractors did not provide basic data. The second example treats an industry, leather tanning, for which the basic data were provided by another contractor. The two methods differed somewhat — particularly in the definition of model plant subcategories and in the way the subcategories are aggregated.

1. EXAMPLE OF COST CALCULATIONS — THE PHENOLIC RESIN INDUSTRY SEGMENT

This calculation of costs begins with data on the industry segment as a whole and proceeds in two parts:

1. Definition of appropriate model plant subcategories, based on available information about plants, waste streams and quantities, treatment and disposal methods, and location that represent the size category under study.
2. Calculation of compliance costs for each of the model plant subcategories.

Once the procedures have been demonstrated, methods of aggregating model plant subcategory costs will be discussed.

a. Model Plant Subcategories

Table H-1 shows data for the large size category phenolic resin model plant. In Part A of Table H-1, the numbers of plants in regulated and nonregulated states are given along with the total quantity of waste generated. Part B describes the waste streams and present practice in disposing of these wastes. Waste stream 1, the aqueous solution, is a major part of the total waste. The management practice section shows the percentage of plants utilizing each disposal technology for that waste stream.

Part C of Table H-1 gives surface impoundment information. These data describe both disposal impoundments and treatment/storage impoundments. Note that treatment/storage impoundments do not appear in Part B under management practices, since they are not disposal technologies. However, they do incur costs that must be included in the compliance costs. Unfortunately, it is not clear from the data what plants have treatment/storage impoundments. The nature of this difficulty will become clear as the model plant subcategories are defined, and its solution will be taken up later.

TABLE H-1
LARGE PHENOLIC RESIN PLANTS MODEL PLANT DATA

A. Plant Information		Number of Plants	Total Waste Generation Per Plant (MT/yr)		
Regulated		22	4800		
Nonregulated		19	4600		

B. Waste Stream Information					
Waste Stream	Total Waste Generation Per Plant (MT/yr)	Monitoring and Testing Code ^a	Waste Management Practices		
1) Aqueous solutions of phenol/formaldehyde	4140	C	Incineration with particulate control 50%	Landfill 35%	Disposal Impoundment 15%
2) Semi-solid waste containing phenol	460	C	Landfill 85%	Disposal Impoundment 15%	

C. Surface Impoundment Information					
State Group	Number of Plants with No Impoundments	Number of Plants with Disposal Impoundments Only	Number of Plants with Treatment/Storage Impoundments Only	Number of Plants with Both Types of Impoundments	Total Number of Plants
Regulated	16	0	3	3	22
Nonregulated	13	0	3	3	19

State Group	Disposal Impoundments		Treatment/Storage Impoundments	
	Number per Plant	Surface Area per Plant (acres)	Number per Plant	Surface Area per Plant (acres)
Regulated	3	23.50	4	100.40
Nonregulated	3	20.50	4	87.30

a. See Appendix B, Table B-3 for explanation.

Source: Arthur D. Little, Inc.

In order to define the model plant subcategories for which compliance costs can be calculated, the management practices used and the waste load going to each practice must be known. It is assumed that each plant will have as few management practices as possible in order to minimize costs. Analysis of Part B of Table H-1 will show that three model plant subcategories are needed to meet the criteria of minimum number of practices and still be consistent with the percentage utilization figures given. These three subcategories are defined in Table H-2. Two of these subcategories (A and C) utilize only one management practice, while B must use two.

TABLE H-2
DEFINITION OF MODEL PLANT
SUBCATEGORIES FOR PHENOLIC RESINS

Subcategory	Disposal Practice Utilized	Load to Disposal Technology (MT/yr)	Percentage of Plants in this Subcategory
A	Sanitary Landfill	4600	35
B	Incineration for Stream ^a	4140	50
	Sanitary Landfill for Stream ^a	460	
C	Disposal Impoundment	4600	15

- a. Stream 1 is the aqueous solution of phenol/formaldehyde shown in Part B of Table H-1.
Stream 2 is the semisolid waste containing phenol.

Source: Arthur D. Little, Inc.

The model plant subcategories defined in Table H-2 do not include treatment/storage impoundments, although Table H-1 indicates that some of them must be present. The data on treatment/storage impoundments do not indicate which of the model plant subcategories utilize such impoundments. This difficulty is handled as follows: costs are calculated for model plant subcategories as defined in Table H-2; costs for treatment/storage impoundments are calculated separately using the subsequent-run set of cost formulas. These costs are then combined with the costs computed for the model plant subcategories to give costs for the model plant. This procedure is demonstrated later.

The next step in calculating costs for each of the model plant subcategories is to define the specific cost parameters required in the cost formulas, based on information in Table H-2. Table H-3 lists and defines the parameters. Table H-4 shows values of the parameters for Subcategory A plants. The only information required to derive these values, beyond that given in Table H-2, concerns the location of the practice (on site or off site).

Whether the plant utilizes on-site or off-site disposal for the management practice in question was determined according to criteria presented in Appendix C

TABLE H-3

COMPLIANCE COST FORMULA NOMENCLATURE

A	Surface impoundment area (acres)
A_W	Average quantity of hazardous waste sent off site per waste stream by a generator (MT/yr) = M/N_W
Cler	Unit wage rate for a clerical worker (\$9.00/hr)
Cont	Cost of installing tank containment (\$)
M	Total hazardous waste generated annually by a model plant (MT/yr)
N_C	Number of different monitoring and testing codes for a model plant
N_G	Number of generators represented by a model plant
N_I	Number of impoundments
N_M	Number of manifests prepared by a model plant
N_{OFF}	Number of generators that dispose of hazardous wastes in off-site facilities
N_{ON}	Number of generators that dispose of hazardous wastes in on-site facilities
N_T	Number of transporters
N_W	Number of hazardous waste streams produced by a model plant
P	Percent of plants at Level 1
Supv	Unit wage rate for a supervisory engineer or research executive (\$27.50/hr)
Tank	Cost of decontaminating and decommissioning hazardous waste storage tank (\$)
Tech _G	Unit wage rate for an engineer employed by a hazardous waste generator (\$29.00/hr)
Tech _{TSDf}	Unit wage rate for an engineer employed by a TSDf (\$35.00/hr)
Truck	Unit wage rate for a truck driver (\$13.50/hr)

Source: Arthur D. Little, Inc.

TABLE H-4
DEFINITION OF COSTING VARIABLES FOR
SUBCATEGORY A PLANTS FOR PHENOLIC RESINS

Costs developed for one plant	N_G	= 1
Two waste streams	N_W	= 2
Both streams monitored by Code C	N_C	= 1
Landfill located off site	N_{ON}	= 0
	N_{OFF}	= 1
Waste load 4600 MT/yr, average load per waste stream 2300; each 20-ton truckload manifested	A_W	= 2300
	$N_M = \frac{4600}{20}$	= 230

Source: Arthur D. Little, Inc.

in which breakpoints for each of the management practices are given. For Subcategory A, the waste load of 4600 MT/yr to landfill falls below the breakpoint; the disposal is off site, as stated in Table H-4.

b. Subcategory Costs

(1) Subcategory A

Based on the parameters shown in Table H-4, the compliance costs for Subcategory A can be calculated using the initial-run formulas. Table H-5 summarizes the results. Table H-6 shows the details of the calculations for Subcategory A. Only one pass through the cost formulas was necessary, since only one management practice was utilized.

Compliance costs depend on whether the plant is located in a regulated or nonregulated state. Costs shown in Table H-5 and later in the example were calculated for plants in nonregulated states. Costs for plants in regulated states are derived by multiplying by a coefficient that depends on cost type and regulation sections. These coefficients are shown in Table H-7.

(2) Subcategory B

The calculation of costs for Subcategory B plants follows the procedure described above. Table H-8 shows the definition of costing parameters, based on information given in Table H-2. Here two management practices are used, landfill for the semi-solid waste, and incineration for the aqueous solution. Therefore, two sets of costing parameters are required, and two runs must be made through the cost formulas. The first run, based on landfill, utilizes the first-run set of cost formulas. The second, for incineration, uses the subsequent-run formulas.

TABLE H-5

PHENOLIC RESINS (SUBCATEGORY A) —
SUMMARY OF COMPLIANCE COSTS FOR A SINGLE MODEL PLANT
LANDFILL — 1 (FIRST RUN)
(\$)

Cost Type	Compliance Activity Requirements						Total
	Treatment/ Disposal	Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	
Capital							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	0	0	0	3,572	0
3005	0	0	0	0	0	0	0
							<u>3,572</u>
Initial							
3001	0	0	800	380	0	0	0
3002	0	34	0	1,025	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	6,028	100	0	66	0
3005	0	0	0	0	0	0	0
							<u>8,433</u>
Annual							
3001	0	0	0	99	0	0	0
3002	0	699	0	1,350	0	0	0
3003	0	0	0	440	0	0	0
3004	51,280	0	0	1,500	0	0	0
3005	0	0	0	0	0	0	0
							<u>55,368</u>
Closure Fund							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	0	0	0	0	3,300
3005	0	0	0	0	0	0	0
							<u>3,300</u>

Source: Arthur D. Little, Inc.

TABLE H-8

PHENOLIC RESINS (SUBCATEGORY A) -
CALCULATION OF COMPLIANCE COST FOR LANDFILL - 1 (FIRST RUN)

Formula Number	Cost Type	RCRA Section	Calculation
a. Treatment/Disposal Requirements			
TR-4	Annual	3004-265.314	$(7.45) M N_{OFF} =$ $(7.45) (4600) (1) =$ 34,270
TR-13	Annual	3004	$N_{OFF} N_W [A_W (50/20) + 800] (1.2985) =$ $1 (2) [(2300) (50/20) + 800] (1.2985) =$ 17,010
NOTE: No other Treatment/Disposal formulas apply.			
b. Recordkeeping and Reporting Requirements			
RR-1	Initial	3002	$N_G (1/2 Cler + Tech_G) =$ $1 [1/2 (9) + 29] =$ 34
RR-2	Annual	3002	$0.05 N_M [1/2 Cler + 1/2 Tech_G] =$ $0.05 (230) [1/2 (9) + 1/2 (29)]$ 219
RR-3	Annual	3002	$N_G (2 Tech_G + Cler) =$ $1 [2 (29) + 9] =$ 67
RR-4	Annual	3002	$110 N_{OFF} =$ 110

TABLE H-6 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
b. Recordkeeping and Reporting Requirements (Continued)			
RR-5	Annual	3002	$N_{OFF} (20 + 5 \text{ Tech}_G + 5 \text{ Supv}) =$ $1 [20 + 5 (29) + 5 (27.5)] =$ 303
RR-6	Annual	3004	$N_{ON} (20 + 12 \text{ Cler}) =$ 0*
RR-7	Annual	3004	$N_{ON} (\text{Cler} + \text{Tech}) =$ 0*
RR-8	Annual	3004	$N_{ON} (24 \text{ Tech}_G + 72 \text{ Cler} + 20) =$ 0*
RR-9	Annual	3004	$N_{ON} (8 \text{ Tech}_G + 3 \text{ Cler} + 20) =$ 0*
RR-10	Annual	3004	NA to disposal mode
RR-11	Initial	3005	$N_{ON} (\text{Tech}_G + 1/2 \text{ Cler}) =$ 0
RR-12	Initial	3005	$N_{ON} (20 \text{ Supv} + 40 \text{ Tech}_G + 15 \text{ Cler}) =$ 0*
c. Monitoring and Testing Requirements			
MT-1	Initial	3001	$400 N_W N_C =$ $400 (2) =$ 800
MT-1a	Initial	3004	$3014 N_G N_W$ $= (3019) (1) (2) = 6028$

*In this example waste is disposed of off-site so N_{ON} is zero, making the total zero.

TABLE H-6 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
c. Monitoring and Testing Requirements (Continued)			
MT-2	Capital	3004	$23,000 N_{ON} =$ 0^*
MT-3	Initial	3004	$3,080 N_{ON} =$ 0^*
MT-4	Annual	3004	$3,400 N_{ON} =$ 0^*
MT-5	Annual	3004	NA to disposal mode
MT-6	Annual	3004	$1,220 N_{ON} =$ 0^*
MT-6a	Initial	3004	NA to disposal mode
MT-7	Annual	3004	NA to disposal mode
d. Administrative Requirements			
AD-1	Initial	3001	$3 N_{OFF} N_W (\text{Cler} + \text{Tech}_G) =$ $3 (1) (2) (9 + 29) =$ 228
AD-2	Initial	3001	$2 N_G N_W (\text{Cler} + \text{Tech}_G) =$ $2 (1) (2) (9 + 29) =$ 152
AD-3	Annual	3001	$2.2 N_G \text{Supv} =$ $2.2 (1) (27.5) =$ 61

TABLE H-6 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
d. Administrative Requirements (Continued)			
AD-4	Annual	3001	$N_G (\text{Cler} + \text{Tech}_G) =$ $(1) (9 + 29) =$ 38
AD-5	Initial	3002	$N_{OFF} (16 \text{ Cler} + 19 \text{ Tech}_G) =$ $1 [16 (9) + 19 (29)] =$ 695
AD-6	Initial	3002	$12 N_{OFF} \text{ Supv} =$ $12 (1) (27.5) =$ 330
AD-7	Annual	3002	$1/3 N_M \text{ Cler} =$ $(1/3) (230) (9) =$ 690
AD-8	Annual	3002	$24 N_{OFF} \text{ Supv} =$ $24 (1) (27.5) =$ 660
AD-9	Annual	3003	$3/4 N_M \text{ Truck} [1/12 + (15/60)(0.15)] =$ $3/4 (230) (13.50) [1/12 + (15/60)(0.15)] =$ 281
AD-10	Annual	3003	$1/4 N_M \text{ Truck} [1/6 + (15/60)(0.15)] =$ $1/4 (230) (13.50) [1/6 + (15/60)(0.15)] =$ 159

TABLE H-6 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
d. Administrative Requirements (Continued)			
AD-11	Initial	3004	$N_{ON} (40 \text{ Tech}_G + 30 \text{ Cler}) =$ 0^*
AD-12	Initial	3004	$N_{ON} (90 \text{ Tech}_G + 32 \text{ Cler}) =$ 0^*
AD-13	Initial	3004	$N_{ON} (35 \text{ Tech}_G + 15 \text{ Cler}) =$ 0^*
AD-14	Initial	3004	$15 N_{ON} (\text{Supv}) =$ 0^*
AD-15	Initial	3004	$100 N_{OFF} =$ $100 (1) =$ 100
AD-16	Annual	3004	$1500 N_{OFF} =$ $1500 (1) =$ 1500
AD-17	Annual	3004	$25 N_{ON} \text{ Tech}_G =$ 0^*
AD-18	Initial	3005	$N_{ON} (40 \text{ Supv} + 20 \text{ Tech}_G + 15 \text{ Cler}) =$ 0^*
e. Contingency Plan Requirements			
CP-1	Capital	3004	$10,000 N_{ON} =$ 0^*
CP-2	Capital	3004	NA to disposal mode
CP-3	Capital	3005	NA to disposal mode

TABLE H-6 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
e. Contingency Plan Requirements (Continued)			
CP-4	Capital	3004	$2 N_{OFF} \text{ Cont} =$ $2 (1) (1,786) =$ 3572
CP-5	Initial	3004	$66 N_{OFF} =$ $66 (1) =$ 66
CP-6	Annual	3004	$(325 + 48 \sqrt{127(M)^{0.651}}) N_{ON}$ 0^*
CP-7	Initial	3005	$N_{ON} (30 \text{ Supv} + 15 \text{ Tech}_G + 10 \text{ Cler}) =$ 0^*
f. Closure Fund Requirements			
CF-1	Closure fund	3004	$500 N_1 =$ 0^*
CF-2	Closure fund	3004	0^*
CF-3	Closure fund	3004	0^*
CF-4	Closure fund	3004	$(2 \text{ Tank} + 100) N_{OFF} =$ $(2 (1,600) + 100) (1) =$ 3,300
CF-5	Closure fund	3004	NA
CF-6	Closure fund	3004	NA
CF-7	Closure fund	3004	NA
CF-8	Closure fund	3004	NA
CF-9	Closure fund	3004	NA

TABLE H-6 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
f. Closure Fund Requirements (Continued)			
CF-10	Closure fund	3004	NA
CF-11	Closure fund	3004	0*
CF-12	Closure fund	3004	NA
CF-13	Closure fund	3004	NA

Source: Arthur D. Little, Inc.

TABLE H-7

STATE REGULATION COST COEFFICIENTS

Activity	Regulation Section				
	3001	3002	3003	3004 (ISS)	3005
Treatment/ Disposal	0.35	0.44	0.3	0.75	0.85
Recordkeeping and Reporting	0.35	0.44	0.3	0.75	0.85
Monitoring and Testing	0.35	0.44	0.3	0.75	0.85
Administrative	0.35	0.44	0.3	0.75	0.85
Training	0.35	0.44	0.3	0.75	0.85
Contingency Plan	0.35	0.44	0.3	0.75	0.85
Closure Fund	1.0	1.0	1.0	1.0	1.0

Source: Arthur D. Little, Inc.

TABLE H-8
DEFINITION OF COSTING VARIABLES
FOR SUBCATEGORY B PLANTS FOR PHENOLIC RESINS

General	
Costs developed for one plant	$N_G = 1$
For Landfill	
Only one waste stream treated	$\begin{cases} N_W = 1 \\ N_C = 1 \end{cases}$
Landfill off site	$\begin{cases} N_{ON} = 0 \\ N_{OFF} = 1 \end{cases}$
Waste load 460 MT/yr originates from one waste stream; each 20-ton truckload manifested	$A_W = 460$ $N_M = \frac{460}{20} = 23$
For Incineration	
Only one waste stream	$\begin{cases} N_W = 1 \\ N_C = 1 \end{cases}$
Incinerator on site	$\begin{cases} N_{ON} = 1 \\ N_{OFF} = 0 \end{cases}$
Waste load 4140 MT/yr originates from one waste stream; waste not manifested	$A_W = 4140$ $N_M = 0$

Sources: Arthur D. Little, Inc.

Tables H-9 and H-10 summarize the results, and Tables H-11 and H-12 give details of these calculations. Adding comparable cost terms from Table H-9 and H-10 gives total compliance costs for a Subcategory B plant.

(3) Subcategory C

Tables H-13 and H-14 show the results for Subcategory C. Table H-13 shows the definitions of costing parameters. Only one management practice is used, and the procedure is essentially the same as for Subcategory A. Table H-15 gives the details of the calculations and summary results.

(4) Treatment/Storage Impoundments

The final calculation that must be made before model plant subcategory and model plant costs can be aggregated is the cost of treatment/storage impoundments. As noted earlier, treatment/storage impoundments cannot be associated with the Subcategory A, B, or C plants, but the associated compliance costs can be calculated. Table H-16 shows the definition of the relevant costing parameters, based on information appearing in Table H-1.

TABLE H-9

PHENOLIC RESINS (SUBCATEGORY B) --
SUMMARY OF COMPLIANCE COSTS FOR A SINGLE MODEL PLANT
SANITARY LANDFILL -- 2 (FIRST RUN)
(\$)

Cost Type	Treatment/ Disposal	Compliance Activity Requirements						Total
		Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	Closure Cost	
Capital								
3001	0	0	0	0	0	0	0	
3002	0	0	0	0	0	0	0	
3003	0	0	0	0	0	0	0	
3004	0	0	0	0	0	2,076	0	
3005	0	0	0	0	0	0	0	<u>2,076</u>
Initial								
3001	0	0	400	190	0	0	0	
3002	0	34	0	1,025	0	0	0	
3003	0	0	0	0	0	0	0	
3004	0	0	3,014	100	0	66	0	
3005	0	0	0	0	0	0	0	<u>4,829</u>
Annual								
3001	0	0	0	99	0	0	0	
3002	0	502	0	729	0	0	0	
3003	0	0	0	44	0	0	0	
3004	5,955	0	0	1,500	0	0	0	
3005	0	0	0	0	0	0	0	<u>8,829</u>
Closure Fund								
3001	0	0	0	0	0	0	0	
3002	0	0	0	0	0	0	0	
3003	0	0	0	0	0	0	0	
3004	0	0	0	0	0	0	0	
3005	0	0	0	0	0	0	2,100	<u>2,100</u>

Source: Arthur D. Little, Inc.

TABLE H-10

**PHENOLIC RESINS (SUBCATEGORY B) -
SUMMARY OF COMPLIANCE COSTS FOR A SINGLE MODEL PLANT
INCINERATION - (SUBSEQUENT RUN)**
(\$)

Cost Type	Compliance Activity Requirements							Total
	Treatment/ Disposal	Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	Closure Fund	
Capital								
3001	0	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0	0
3004	0	0	0	0	0	7,500	0	0
3005	0	0	0	0	0	0	0	0
								<u>7,500</u>
Initial								
3001	0	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0	0
3004	0	0	3,014	0	0	0	0	0
3005	0	464	0	456	0	0	0	0
								<u>3,934</u>
Annual								
3001	0	0	0	0	0	0	0	0
3002	0	67	0	0	0	0	0	0
3003	0	0	0	0	0	0	0	0
3004	0	1,270	11,000	145	0	0	0	0
3005	0	0	0	0	0	0	0	0
								<u>12,482</u>
Closure Fund								
3001	0	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0	0
3004	0	0	0	0	0	0	204,120	0
3005	0	0	0	0	0	0	0	0
								<u>204,120</u>

Sources: Arthur D. Little, Inc.

TABLE H-11
PHENOLIC RESINS (SUBCATEGORY B) -
CALCULATION OF COMPLIANCE COSTS FOR LANDFILL - 2 (FIRST RUN)

Formula Number	Cost Type	RCRA Section	Calculation
a. Treatment/Disposal Requirements			
TR-4	Annual	3004	$(7.45) MN_{OFF} =$ $(7.45) (460) (1) =$ 3,427
TR-13	Annual	3004	$N_{OFF} N_W [A_W (50/20) + 800] (1.2985) =$ $(1) (1) [(458.7) (50/20) + 800] (1.2985) =$ 2,528
b. Recordkeeping and Reporting Requirements			
RR-1	Initial	3002	$N_G (1/2 \text{ Cler} + \text{Tech}_G) =$ $(1) [1/2 (9) + 29] =$ 34
RR-2	Annual	3002	$0.05 N_M (1/2 \text{ Cler} + 1/2 \text{Tech}_G) =$ $0.05 (23) [1/2 (9) + 1/2 (29)] =$ 22
RR-3	Annual	3002	$N_G (2 \text{Tech}_G + \text{Cler}) =$ $(1) [2 (29) + 9] =$ 67
RR-4	Annual	3002	$110 N_{OFF} =$ $110 (1) =$ 110
RR-5	Annual	3002	$N_{OFF} (20 + 5 \text{Tech}_G + 5 \text{Supv}) =$ $(1) [20 + 5 (29) + 5 (27.5)] =$ 303
RR-6	Annual	3004	$N_{ON} (20 + 12 \text{Cler}) =$ 0*

*In this example waste is disposed of off-site so N_{ON} is zero making the total zero.

TABLE H-11 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
b. Recordkeeping and Reporting Requirements (Continued)			
RR-7	Annual	3004	$N_{ON} (\text{Cler} + \text{Tech}_G) =$ 0^*
RR-8	Annual	3004	$N_{ON} (24 \text{ Tech}_G + 72 \text{ Cler} + 20) =$ 0^*
RR-9	Annual	3004	$N_{ON} (8 \text{ Tech}_G + 3 \text{ Cler}) =$ 0^*
RR-10	Annual	3004	NA to disposal mode
RR-11	Initial	3005	$N_{ON} (\text{Tech}_G + 1/2 \text{ Cler} + 20) =$ 0^*
RR-12	Initial	3005	$N_{ON} (20 \text{ Supv} + 40 \text{ Tech}_G + 15 \text{ Cler}) =$ 0^*
c. Monitoring and Testing Requirements			
MT-1	Initial	3001	$400 N_W N_C =$ $400 (1) (1) =$ 400
MT-1a	Initial	3004	$3,014 N_G N_W =$ $3,014 (1) (1) =$ 3,014
MT-2	Capital	3004	$23,000 N_{ON} =$ 0^*
MT-3	Initial	3004	$3,080 N_{ON} =$ 0^*
MT-4	Annual	3004	$3,400 N_{ON} =$ 0^*
MT-5	Annual	3004	NA to disposal mode

TABLE H-11 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
c. Monitoring and Testing Requirements (Continued)			
MT-6	Annual	3004	$1,220 N_{ON} =$ 0^*
MT-6a	Initial	3004	NA to disposal mode
MT-7	Annual	3004	NA to disposal mode
d. Administrative Requirements			
AD-1	Initial	3001	$3 N_{OFF} N_W (\text{Cler} + \text{Tech}_G) =$ $3 (1) (1) (29 + 9) =$ 114
AD-2	Initial	3001	$2 N_G N_W (\text{Cler} + \text{Tech}_G) =$ $2 (1) (1) (9 + 29) =$ 78
AD-3	Annual	3001	$2.2 N_G N_C \text{ Supv} =$ $2.2 (1) (1) (27.5) =$ 61
AD-4	Annual	3001	$N_W N_G (\text{Cler} + \text{Tech}_G) =$ $(1) (1) (9 + 29) =$ 38
AD-5	Initial	3002	$N_{OFF} (16 \text{ Cler} + 19 \text{ Tech}_G) =$ $1 [16 (9) + 19 (29)] =$ 695
AD-6	Initial	3002	$12 N_{OFF} \text{ Supv} =$ $12 (1) (27.5) =$ 330
AD-7	Annual	3002	$1/3 N_M \text{ Cler} =$ $(1/3) (23) (9) =$ 69

TABLE H-11 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
d. Administrative Requirements (Continued)			
AD-8	Annual	3002	$24 N_{OFF} \text{ Supv}$ $24 (1) (27.5) =$ 660
AD-9	Annual	3003	$3/4 N_M \text{ Truck } [1/12 + (15/60)(0.15)]$ $3/4 (22.9) (13.50) [1/12 + (15/60)(0.15)] =$ 28
AD-10	Annual	3003	$1/4 N_M \text{ Truck } [1/6 + (15/60)(0.15)] =$ $1/4 (22.9) (13.50) [1/6 + (15/60)(0.15)] =$ 16
AD-11	Initial	3004	$N_{ON} (40 \text{ Tech}_G + 30 \text{ Cler}) =$ 0*
AD-12	Initial	3004	$N_{ON} (90 \text{ Tech}_G + 32 \text{ Cler}) =$ 0*
AD-13	Initial	3004	$N_{ON} (35 \text{ Tech}_G + 15 \text{ Cler}) =$ 0*
AD-14	Initial	3004	$15 N_{ON} \text{ Supv} =$ 0*
AD-15	Initial	3004	$100 N_{OFF} =$ $100 (1) =$ 100
AD-16	Annual	3004	$1,500 N_{OFF} =$ $1,500 (1) =$ 1,500
AD-17	Annual	3004	$25 N_{ON} \text{ Tech}_G =$ 0*
AD-18	Initial	3005	$N_{ON} (40 \text{ Supv} + 20 \text{ Tech} + 15 \text{ Cler}) =$ 0*

TABLE H-11 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
e. Contingency Plan Requirements			
CP-1	Capital	3004	$10,000 N_{ON} =$ 0^*
CP-2	Capital	3004	NA to disposal mode
CP-3	Capital	3004	NA to disposal mode
CP-4	Capital	3004	$2 N_{OFF} \text{ Cont} =$ $2 (1) (1,038) =$ $2,076$
CP-5	Initial	3004	$66 N_{OFF} =$ $66 (1) =$ 66
CP-6	Annual	3004	NA to disposal mode
CP-7	Initial	3005	$N_{ON} (30 \text{ Supv}_G + 15 \text{ Tech}_G + 10 \text{ Cler}_{TSD}) =$ 0^*
f. Closure Fund Requirements			
CF-1	Closure fund	3004	NA
CF-2	Closure fund	3004	NA
CF-3	Closure fund	3004	NA
CF-4	Closure fund	3004	$(2 \text{ Tank} + 100) N_{OFF} = 2 (1000) + 100 =$ 2100
CF-5	Closure fund	3004	NA
CF-6	Closure fund	3004	NA
CF-7	Closure fund	3004	NA
CF-8	Closure fund	3004	NA
CF-9	Closure fund	3004	NA
CF-10	Closure fund	3004	NA
CF-11	Closure fund	3004	NA

TABLE H-11 (Continued)

Formula Number	Cost Type	RCRA Section	Calculation
f. Closure Fund Requirements (Continued)			
CF-12	Closure fund	3004	NA
CF-13	Closure fund	3004	NA
CF-14	Closure fund	3004	$(2 \text{ Tank} + 100) N_{OFF} =$ $(2 \text{ 1,000} + 100) (1) =$ 2,100

Source: Arthur D. Little, Inc.

TABLE H-12

**PHENOLIC RESINS (SUBCATEGORY B) –
CALCULATION OF COMPLIANCE COSTS FOR INCINERATION (SUBSEQUENT RUN)**

a. Recordkeeping and Reporting Requirements

Formula Number	Cost Type	RCRA Section	Calculation
I-RR-1	Annual	3002	$0.05 N_M (1/2 \text{ Cler} + 1/2 \text{ Tech}_G) =$ 0^*
I-RR-2	Annual	3002	$N_G (2 \text{ Tech}_G + \text{Cler}) =$ $(1) [2 (29) + 9] =$ 67
I-RR-3	Annual	3004	$N_{ON} (\text{Cler} + \text{Tech}_G) =$ $(1) (9 + 29) =$ 38
I-RR-4	Annual	3004	$N_{ON} (12 \text{ Tech}_G + 24 \text{ Cler} + 20) =$ $(1) [12 (29) + 24 (9) + 20] =$ 584
I-RR-5	Annual	3004	NA to disposal mode
I-RR-6	Annual	3004	$2 N_{ON} (8 \text{ Tech}_G + 2 \text{ Supv} + 3 \text{ Cler} + 20) =$ $2 (1) [8 (29) + 2 (27.5) + 3 (9) + 20] =$ 648
I-RR-7	Initial	3005	$N_{ON} (5 \text{ Supv}_G + 10 \text{ Tech}_G + \text{Cler}) =$ $(1) [5 (27.5) + 10 (29) + 4 (9)] =$ 464

*In this example the waste is incinerated on-site. There are therefore no manifests, making N_M zero and the total zero.

TABLE H-12 (Continued)

b. Monitoring and Testing Requirements

Formula Number	Cost Type	RCRA Section	Calculation
I-MT-1	Capital	3001	NA to disposal mode
I-MT-1a	Initial	3001	3014
I-MT-2	Initial	3004	NA to disposal mode
I-MT-3	Annual	3004	NA to disposal mode
I-MT-4	Annual	3004	$11,000 N_{ON} =$ $11,000 (1) =$ 11,000
I-MT-5	Annual	3004	NA to disposal mode
I-MT-6	Initial	3004	NA to disposal mode
I-MT-7	Annual	3004	NA to disposal mode

c. Administrative Requirements

I-AD-1	Annual	3002	$1/3 N_M \text{ Cler} =$ 0°
I-AD-2	Annual	3003	NA
I-AD-3	Annual	3003	NA
I-AD-4	Annual	3004	$5 N_{ON} \text{ Tech}_G =$ $5 (1) (29) =$ 145
I-AD-5	Initial	3005	$N_{ON} (10 \text{ Supv} + 5 \text{ Tech}_G + 4 \text{ Cler}) =$ $(1) [10 (27.5) + 5 (29) + 4 (9)] =$ 456

TABLE H-12 (Continued)

d. Contingency Plan Requirements

Formula Number	Cost Type	RCRA Section	Calculation
I-CP-1	Capital	3004	7,500 N_{ON} = 7,500 (1) = 7,500
I-CP-2	Capital	3004	NA to disposal mode
I-CP-3	Capital	3004	NA to disposal mode
I-CP-4	Annual	3004	NA to disposal mode

e. Closure Fund Requirements

I-CF-1	Closure fund	3004	500
I-CF-2	Closure fund	3004	NA to disposal mode
I-CF-3	Closure fund	3004	NA to disposal mode
I-CF-4	Closure fund	3004	NA to disposal mode
I-CF-5	Closure fund	3004	NA to disposal mode
I-CF-6	Closure fund	3004	202,000
I-CF-7	Closure fund	3004	1,620
I-CF-8	Closure fund	3004	NA to disposal mode
I-CF-9	Closure fund	3004	NA to disposal mode
I-CF-10	Closure fund	3004	NA to disposal mode
I-CF-11	Closure fund	3004	NA to disposal mode
I-CF-12	Closure fund	3004	NA to disposal mode

Sources: Arthur D. Little, Inc.

TABLE H-13

DEFINITION OF COSTING VARIABLES
FOR SUBCATEGORY C PLANTS FOR PHENOLIC RESINS

Costs developed for one plant	$N_G = 1$
Two waste streams	$N_W = 2$
Both streams monitored by Code C	$N_C = 1$
Impoundments located on site	$\left\{ \begin{array}{l} N_{ON} = 1 \\ N_{OFF} = 0 \end{array} \right.$
Three impoundments per plant	$N_I = 3$
Surface area of impoundment	
7.83 acres in Regulated states	$A = 7.83$
and 6.83 acres in Nonregulated states	or $A = 3.83$

Source: Arthur D. Little, Inc.

TABLE H-14

PHENOLIC RESINS (SUBCATEGORY C) - SUMMARY OF COMPLIANCE COSTS
FOR A SINGLE MODEL PLANT - DISPOSAL IMPOUNDMENTS (FIRST RUN)
(\$)

Cost Type	Compliance Activity Requirements						Contingency Plan	Closure Fund	Total
	Treatment/ Disposal	Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training				
Capital									
3001	0	0	0	0	0		0	0	
3002	0	0	0	0	0		0	0	
3003	0	0	0	0	0		0	0	
3004-Regulated	0	0	23,000	0	0		54,893	0	
3004-Nonregulated	0	0	23,000	0	0		52,244	0	
3005	0	0	0	0	0		0	0	
Initial									
3001	0	0	800	152	0		0	0	77,893
3002	0	34	0	0	0		0	0	75,244
3003	0	0	0	0	0		0	0	
3004	0	0	9,108	5,891	10,175		0	0	
3005	0	1,878	0	1,815	0		1,360	0	
Annual									
3001	0	0	0	89	0		0	0	
3002	0	67	0	0	0		0	0	
3003	0	0	0	0	0		0	0	
3004	0	1,808	4,620	725	4,500		0	0	
3005	0	0	0	0	0		0	0	
Closure Fund									
3001	0	0	0	0	0		0	0	
3002	0	0	0	0	0		0	0	
3003	0	0	0	0	0		0	0	
3004-Regulated	0	0	0	0	0		1,961,285	0	
3004-Nonregulated	0	0	0	0	0		1,768,811	0	
3005	0	0	0	0	0		0	0	
Total									
									1,961,285
									1,768,811

TABLE H-15

**PHENOLIC RESINS (SUBCATEGORY C)
CALCULATION OF COMPLIANCE COSTS FOR
DISPOSAL IMPOUNDMENTS (FIRST RUN)**

Formula Number	Cost Type	RCRA Section	Cost (\$)
a. Treatment/Disposal Requirements			
TR-1	Capital	3004	NA
TR-2	Annual	3004	NA
TR-3	Annual	3004	NA
TR-4	Annual	3004	NA
TR-5	Annual	3004	NA
TR-6	Annual	3004	NA
TR-7	Annual	3004	NA
TR-8	Annual	3004	NA
TR-9	Annual	3004	NA
TR-10	Annual	3004	NA
b. Recordkeeping and Reporting Requirements			
RR-1	Initial	3002	34
RR-2	Annual	3002	0
RR-3	Annual	3002	67
RR-4	Annual	3002	0
RR-5	Annual	3002	0
RR-6	Annual	3004	128
RR-7	Annual	3004	38
RR-8	Annual	3004	1,364
RR-9	Annual	3004	279
RR-10	Annual	3004	0
RR-11	Initial	3005	34
RR-12	Initial	3005	1,845

TABLE H-15 (Continued)

Formula Number	Cost Type	RCRA Section	Cost (\$)
c. Monitoring and Testing Requirements			
MT-1	Initial	3001	800
MT-1a	Initial	3004	6,028
MT-2	Capital	3004	23,000
MT-3	Initial	3004	3,080
MT-4	Annual	3004	3,400
MT-5	Annual	3004	NA
MT-6	Annual	3004	1,220
MT-6a	Initial	3004	NA
MT-7	Annual	3004	NA
d. Administrative Requirements			
AD-1	Initial	3001	0
AD-2	Initial	3001	152
AD-3	Annual	3001	61
AD-4	Annual	3001	38
AD-5	Initial	3002	0
AD-6	Initial	3002	0
AD-7	Annual	3002	0
AD-8	Annual	2002	0
AD-9	Annual	3003	0
AD-10	Annual	3003	0
AD-11	Initial	3004	1,430
AD-12	Initial	3004	2,898
AD-13	Initial	3004	1,150
AD-14	Initial	3004	413
AD-15	Initial	3004	0
AD-16	Annual	3004	0
AD-17	Annual	3004	725
AD-18	Initial	3005	1,815

TABLE H-15 (Continued)

Formula Number	Cost Type	RCRA Section	Cost (\$)
e. Training Requirements			
TG-1	Initial	3004	10,175
TG-2	Annual	3004	NA
TG-3	Annual	3004	4,500
f. Contingency Plan Requirements			
CP-1	Capital	3004	10,000
CP-2	Capital	3004	NA
CP-3	Capital	3004	Regulated 44,893
	Capital	3004	Nonregulated 42,244
CP-4	Capital	3004	0
CP-5	Initial	3004	0
CP-6	Annual	3004	NA
CP-7	Initial	3005	1,350
g. Closure Fund Requirements			
CF-1	Closure fund	3004	1,500
CF-2	Closure fund	3004	NA
CF-3	Closure fund	3004	NA
CF-4	Closure fund	3004	0
CF-5	Closure fund	3004	NA
CF-6	Closure fund	3004	NA
CF-7	Closure fund	3004	NA
CF-8	Closure fund	3004	NA
CF-9	Closure fund	3004	8,250
CF-10	Closure fund	3004	393,300
CF-11	Closure fund	3004	NA
CF-12	Closure fund	3004	0
CF-13	Closure fund	3004	Regulated 1,558,591
	Closure fund	3004	Nonregulated 1,365,114

Source: Arthur D. Little, Inc.

TABLE H-16

**DEFINITION OF COSTING VARIABLES
FOR TREATMENT/STORAGE IMPOUNDMENTS FOR PHENOLIC RESINS**

Costs developed for one plant	$N_G = 1$
Impoundments located on site	$\left\{ \begin{array}{l} N_{ON} = 1 \\ N_{OFF} = 0 \end{array} \right.$
Four impoundments per plant	$N_I = 4$
Surface area of impoundment	
25.1 acres in Regulated states and	$A = 25.1$
21.8 in Nonregulated states	or $A = 21.8$

Note: Values for N_W and N_C not required to determine cost of practices.

Source: Arthur D. Little, Inc.

Any plant having treatment/storage impoundments must also utilize some disposal practice. Therefore, compliance costs for treatment/storage impoundments are calculated using the subsequent-run cost formulas. Table H-17 summarizes the costs.

c. Model Plant Costs — Phenolic Resins Example

Model plant subcategory compliance costs are aggregated to estimate model plant compliance costs. Model plant costs have been developed for plants without surface impoundments as well as for those with various combinations of surface impoundments to show the significance of the cost of compliance for surface impoundments. Aggregate costs of compliance are then calculated for plants in the large size category in nonregulated states.

(1) Model Plant without Impoundments

Compliance costs for the model plant are simply the weighted average of the costs of all subcategory plants that do not utilize surface impoundments. In the above example, these plants would include Subcategories A and B, but not Subcategory C.

Table H-2 shows that 35% of the plants are Subcategory A and 50% are Subcategory B. Therefore, the cost for this model plant would be

$$C_1 = (0.35/(0.35 + 0.50)) C_A + (0.5/(0.35 + 0.50)) C_B$$

where:

C_A is the cost element for Subcategories A shown in Table H-5, and
 C_B is the sum of the cost element for the two management practices in Subcategory B (Tables H-9 and H-10).

TABLE H-17

PHENOLIC RESINS - SUMMARY OF
COMPLIANCE COSTS FOR A SINGLE MODEL PLANT - TREATMENT/STORAGE IMPOUNDMENTS
(\$)

Cost Type	Compliance Activity Requirements						Total
	Treatment/ Disposal	Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	
Capital							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004-Regulated	0	0	23,000	0	0	108,159	0
3004-Nonregulated	0	0	23,000	0	0	102,010	0
3005	0	0	0	0	0	0	0
							<u>131,159</u>
							<u>125,010</u>
Initial							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	3,080	0	0	0	0
3005	0	484	0	456	0	0	0
							<u>4,000</u>
Annual							
3001	0	0	0	0	0	0	0
3002	0	67	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004	0	801	4,620	146	0	0	0
3005	0	0	0	0	0	0	0
							<u>5,733</u>
Closure Fund							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004-Regulated	0	0	0	0	0	0	6,971,907
3004-Nonregulated	0	0	0	0	0	0	6,155,581
3005	0	0	0	0	0	0	0
							<u>6,971,907</u>
							<u>6,155,581</u>

Source: Arthur D. Little, Inc.

Table H-18 summarizes the costs so calculated for the model plant without impoundments.

(2) Model Plant with Treatment/Storage Impoundments Only

Table H-1 shows that there are six plants with treatment/storage impoundments and no disposal impoundments. Costs for this type of model plant would be

$$C_2 = C_1 + C_{TSI}$$

where:

C_1 is the cost of a model plant with no impoundments (see above) and C_{TSI} is the cost of treatment/storage impoundments shown in Table H-17.

Table H-18 summarizes the costs obtained.

(3) Model Plant with Disposal Impoundments Only

Table H-1 shows that, for large plants in the phenolic resin industry segment, there are no plants in the category of model plants with disposal impoundments only. The costs associated with such a plant would be only the costs of model plant Subcategory C, as shown in Table H-14, since this is the only plant that utilizes disposal impoundments, or

$$C_3 = C_C$$

(4) Model Plant with Disposal and Treatment/Storage Impoundments

Table H-1 shows that there are six plants with both types of impoundments in the example industry. The cost elements would be:

$$C_4 = C_3 + C_{TSI}$$

where:

C_3 is the cost element for model plants with disposal impoundments only (see above) and

C_{TSI} is the cost of treatment/storage impoundments shown in Table H-17. Table H-19 summarizes the costs obtained.

The costs listed here are associated with plants located in nonregulated states. Comparable cost figures for plants located in regulated states can be obtained by multiplying the given cost elements associated with the basic model plant subcategories by the aggregate coefficient as shown earlier in Table H-7.

TABLE H-18

PHENOLIC RESINS - SUMMARY OF COMPLIANCE COSTS FOR A SINGLE MODEL - PLANT WITHOUT IMPOUNDMENTS
(\$)

Cost Type	Treatment/ Disposal	Compliance Activity Requirements					Total
		Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	
Capital							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	0	0	0	7,103	0
3005	0	0	0	0	0	0	0
							7,103
Initial							
3001	0	0	565	268	0	0	0
3002	0	34	0	1,025	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	8,028	100	0	68	0
3005	0	273	0	268	0	0	0
							8,627
Annual							
3001	0	0	0	89	0	0	0
3002	0	622	0	985	0	0	0
3003	0	0	0	207	0	0	0
3004	24,618	747	6,471	1,585	0	0	0
3005	0	0	0	0	0	0	0
							35,334
Closure Fund							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	0	0	0	0	0
3005	0	0	0	0	0	0	0
							122,665
							122,665

TABLE H-19

**PHENOLIC RESINS - SUMMARY OF COMPLIANCE COSTS FOR A SINGLE MODEL
PLANT - WITH TREATMENT/STORAGE IMPOUNDMENTS ONLY**
(\$)

Cost Type	Compliance Activity Requirements						Total
	Treatment/ Disposal	Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	
Capital							
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004-Regulated	0	0	23,000	0	0	116,262	0
3004-Nonregulated	0	0	23,000	0	0	109,113	0
3005	0	0	0	0	0	0	0
Initial							138,262
3001	0	0	565	268	0	0	0
3002	0	34	0	1,025	0	0	0
3003	0	0	0	0	0	0	0
3004	0	0	8,108	100	0	66	0
3005	0	737	0	724	0	0	0
Annual							12,627
3001	0	0	0	99	0	0	0
3002	0	689	0	885	0	0	0
3003	0	0	0	207	0	0	0
3004	24,618	1,848	10,321	1,730	0	0	0
3005	0	0	0	0	0	0	0
Closure Fund							40,297
3001	0	0	0	0	0	0	0
3002	0	0	0	0	0	0	0
3003	0	0	0	0	0	0	0
3004-Regulated	0	0	0	0	0	0	7,094,572
3004-Nonregulated	0	0	0	0	0	0	6,278,246
3005	0	0	0	0	0	0	0
							7,094,572
							6,278,246

Source: Arthur D. Little, Inc.

(3) *Model Plant Cost for the Large Size Category for a Nonregulated State*

Costs for the model plant subcategory are tabulated in Table H-20 to facilitate understanding of the computation analysis. Table H-1 showed 19 plants in non-regulated states, six of them with treatment/storage impoundments. Table H-2 showed that there are three plants with disposal impoundments (Subcategory C), and that of the remaining 16 plants, the number in Subcategory A is:

$$N_A = (16) [(0.35)/(0.35 + 0.50)] = 6.5882$$

and the number in Subcategory B is:

$$N_B = (16) [(0.50)/(0.35 + 0.50)] = 9.4118$$

In addition, the number of these plants with treatment/disposal impoundments is $N_2 = 6$. The formula for calculating the model plant cost for each cost type is:

$$(N_A C_A + N_B C_B + N_C C_C + N_2 C_{T1}) / (N_A + N_B + N_C)$$

where:

N_A, N_B, N_C are the number of plants in each subcategory
 C_A, C_B, C_C are the cost type for each subcategory
 C_{T1} is the cost type for treatment/storage impoundments

The model plant costs and total model plant cost for the large, nonregulated size category are presented in Table H-21 and H-22.

The rounded-off results of the hand-calculated sample and the computer calculations for the large size and nonregulated state category for the phenolic resin segment are:

	Capital	Initial	Annual	Closure
			(\$000)	
Hand-calculated	1,089	256	636	44,203
Computer-calculated	1,096	261	635	44,210

The hand-calculated and computer-calculated costs for all cost types are in good agreement; small differences are caused by rounding and minor differences in the formulas used in the hand and computer calculations.

d. **Annualized Total Cost**

The annualized cost is calculated by converting 20 years of compliance costs to an annuity with 20 equal yearly payments. The present value of the annuity equals the present value of the compliance costs. Inflation is incorporated in the compliance costs.

TABLE H-20

**PHENOLIC RESINS - SUMMARY OF COMPLIANCE COSTS
FOR A SINGLE MODEL PLANT - WITH BOTH TYPES OF IMPOUNDMENTS**
(5)

Cost Type	Compliance Cost Activities							Total
	Treatment/ Disposal	Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training	Contingency Plan	Closure Fund	
Capital								
3001	0	0	0	0	0	0	0	
3002	0	0	0	0	0	0	0	
3003	0	0	0	0	0	0	0	
3004-Regulated	0	0	46,000	0	0	163,052	0	
3004-Nonregulated	0	0	46,000	0	0	154,254	0	
3005	0	0	0	0	0	0	0	<u>209,052</u>
Initial								<u>200,254</u>
3001	0	0	800	152	0	0	0	
3002	0	34	0	0	0	0	0	
3003	0	0	0	0	0	0	0	
3004	0	0	12,188	5,881	10,176	0	0	
3005	0	2,343	0	2,271	0	1,350	0	<u>36,204</u>
Annual								
3001	0	0	0	89	0	0	0	
3002	0	134	0	0	0	0	0	
3003	0	0	0	0	0	0	0	
3004	0	2,710	9,240	870	4,500	0	0	
3005	0	0	0	0	0	0	0	<u>17,553</u>
Closure Fund								
3001	0	0	0	0	0	0	0	
3002	0	0	0	0	0	0	0	
3003	0	0	0	0	0	0	0	
3004-Regulated	0	0	0	0	0	0	8,933,192	
3004-Nonregulated	0	0	0	0	0	0	7,924,392	
3005	0	0	0	0	0	0	0	<u>8,933,192</u>
								<u>7,924,392</u>

Source: Arthur D. Little, Inc.

TABLE H-21

SUMMARY OF MODEL PLANT SUBCATEGORY COSTS
FOR PHENOLIC RESINS^a

(\$)

Subcategory	Capital	Initial	Annual	Closure Fund
A	3,572	8,433	55,368	3,300
B	9,576	8,763	21,311	206,220
C	75,244	31,204	11,820	1,768,811
TSI	125,010	4,000	5,733	6,155,581

a. Hand-calculated values

Source: Arthur D. Little, Inc.

TABLE H-22

SUMMARY OF MODEL PLANT COSTS FOR PHENOLIC RESINS^a

(\$)

Cost Type	Model Plant Cost	Size Category Cost
Capital	57,340	1,089,452
Initial	13,455	255,646
Annual	33,453	635,604
Closure Fund	2,326,451	44,202,561

a. Hand-calculated values

Source: Arthur D. Little, Inc.

In determining total annualized cost, the following assumptions were made:

- The inflation rate is 10% for the first 5 years and 7% for the remaining 15 years.
- The cost of capital is 18.5% for the first 5 years and 15.7% for the remaining 15 years.

Capital expenditures and initial costs are incurred in the first year. It was assumed that these costs were made in the first six months of the year. A minor adjustment was made to reflect the net present value of these expenditures. The annual cost is projected over 20 years, increasing with inflation. The closure fund contributions are made in the first five years only, inflating during that period.* The formulas to calculate these costs are presented in Table H-23.

Table H-24 gives the numerical factors used to calculate annualized costs for the large size, nonregulated category for phenolic resins. The total annualized cost of \$6,450,038 is presented in Table H-25; the comparable computer-calculated value is \$7,832,406.

e. Total Industry Segment Costs

The calculation of total industry segment costs requires three additional sets of costs comparable to those shown in Table H-25. The three sets are associated with:

- Large plants in regulated states — These values are derived from those calculated above and summarized in Table H-25 by application of the cost coefficients shown earlier in Table H-7.
- Medium size category plants in nonregulated states — These values are calculated exactly as shown above using data for plants in the medium size range.
- Medium size plants in regulated states — These values are derived as above from the nonregulated-state values.

The total industry costs are obtained by adding the comparable costs associated with the four plant-size and state location groups.

2. A SECOND EXAMPLE — LEATHER TANNING

For the industries for which basic data were supplied to Arthur D. Little, Inc., by other contractors, the methodology for calculating compliance costs differed somewhat from that demonstrated in the preceding example. The important differences occur in the definition of model plant subcategories and the way in which

*Subsequent to preparing this set of calculations the closure fund trust fund payment period was changed from five years to twenty years. While this sample calculation used a five-year period for accumulation of the closure fund, the formulas in the methodology chapter and numerical values throughout the rest of this report are based on the 20-year scheme.

TABLE H-23

FORMULAS FOR ANNUALIZING COSTS

Capital or Initial Cost

$$\text{Annualized Cost} = (\text{Initial or Capital Cost}) \left[\frac{1 + \frac{\text{Inf}_1}{2}}{1 + \frac{\text{CF}_1}{2}} \right] (\text{Annuity factor})$$

$$= 0.1743 (\text{Initial or Capital Cost})$$

Annual

$$\text{Annualized Cost} = \text{Annual Cost} \left[\sum_{n=1}^5 \left(\frac{1 + \text{Inf}_1}{1 + \text{CF}_1} \right)^n + \left(\frac{1 + \text{Inf}_1}{1 + \text{CF}_1} \right)^5 \sum_{n=1}^{15} \left(\frac{1 + \text{Inf}_2}{1 + \text{CF}_2} \right)^n \right] (\text{Annuity factor})$$

$$= 1.791 (\text{Annual Cost})$$

Closure Fund

$$\text{Annualized Cost} = \frac{\text{Closure Fund}}{y} \left[\sum_{n=1}^5 \left(\frac{1 + \text{Inf}_1}{1 + \text{CF}_1} \right)^n + \left(\frac{1 + \text{Inf}_1}{1 + \text{CF}_1} \right)^5 \sum_{n=1}^{15} \left(\frac{1 + \text{Inf}_2}{1 + \text{CF}_2} \right)^n \right] (\text{Annuity factor})$$

y = years over which fund is built up

For y = 20 yrs

$$\text{Annualized Cost} = 1.791 \left(\frac{\text{Closure Fund}}{20} \right)$$

Annuity Factor

$$\text{Annuity Factor} = \left[\sum_{n=1}^5 \left(\frac{1}{1 + \text{CF}_1} \right)^n + \left(\frac{1}{1 + \text{CF}_1} \right)^5 \sum_{n=1}^{15} \left(\frac{1}{1 + \text{CF}_2} \right)^n \right]^{-1} = 0.1814$$

Inf₁ = 0.10 (years 1-5)Inf₂ = 0.07 (years 6-20)CF₁ = 0.185 (years 1-5)CF₂ = 0.157 (years 6-20)

TABLE H-24

ANNUALIZATION FACTORS TO CALCULATE ANNUALIZED COSTS

1. Capital Expenditures

Discount Factor

$$\left(\frac{1.05}{1.0925} \right)$$

Annuity Factor

(0.1814)

=

Annualization Factor

0.1743

2. Initial Expenses

$$\left(\frac{1.05}{1.0925} \right)$$

(0.1814)

=

0.1743

3. Annual Operating Cost

Year Discount Factor

1	0.928
2	0.862
3	0.800
4	0.743
5	0.689
6	0.637
7	0.589
8	0.545
9	0.504
10	0.466
11	0.431
12	0.399
13	0.369
14	0.341
15	0.315
16	0.292
17	0.270
18	0.249
19	0.231
20	0.213

(9.873)

(0.1814)

=

1.791

4. Closure Fund Payment

Year Discount Factor

1	.928
2	.862
3	.800
4	.743
5	.689

(4.022)

(0.1814)

=

0.7296

TABLE H-25
TOTAL ANNUALIZED COMPLIANCE COSTS FOR
LARGE, NONREGULATED SIZE CATEGORY OF PHENOLIC RESINS
(\$)

Capital Expenditures		Annualization Factor		Total Annualized Cost
1,089,452	x	0.1814	=	197,827
Initial Expenses				
255,646	x	0.1814	=	46,374
Annual Operating Costs				
635,604	x	1.791	=	1,138,367
Closure Fund Payment				
44,202,561	x	0.7296	=	6,450,038
5				7,832,406

Source: Arthur D. Little, Inc.

subcategory costs are aggregated to give model plant costs. The following example, for the small size category of the nonchrome leather tanning segment of the leather tanning and finishing industry, shows the differences in methodology. Where the methodology is the same as that shown in the preceding example, it is not repeated in detail.

Table H-26 shows the basic data supplied to Arthur D. Little, Inc., by Pope-Reid Associates, Inc., on which the calculations for the small size category of the nonchrome leather tanning segment were based. Model plant subcategories are defined directly in terms of waste streams, waste loads, and management practices utilized; the more complicated definition procedure of the preceding example is not needed.

As before, the first step in calculating compliance costs is to translate the descriptions of model plant subcategories into costing parameters used in the calculations. Only the calculations for Subcategory B are followed here, since the applications to the other subcategories should be apparent.

Subcategory B utilizes two disposal practices (landfill and disposal impoundments) and also has treatment/storage impoundments. Therefore, as described previously, three runs through the cost formulas are required. Table H-27 shows the parameters used for each run.

The first-run cost formulas were used to calculate costs associated with the landfill. Table H-28 summarizes the results. Table H-29 gives the details.

TABLE H-26
MODEL PLANT DATA FOR LEATHER TANNING AND FINISHING -
NONCHROME TAN - SMALL
(Production: 531 MT/yr)

Subcategory	Waste Streams	Quantity (MT/yr)	Number of Plants	Waste Management Technology	Treatment/Storage Impoundments	
					Number per Plant	Surface Area per Impoundment (acres)
A	1. Screenings	13	1	1. On-site Landfill 2. Disposal Impoundment (1 Impoundment, 0.33 acres)	-	-
	2. Sludge	1146				
B	1. Screenings	13	1	1. Off-site Landfill 2. Disposal Impoundment (1 Impoundment, 0.33 acres)	2	0.5
	2. Sludge	1146				
C	1. Screenings	13	2	1. Off-site Landfill 2. Off-site Landfill	-	-
	2. Sludge	191				
D	None	-	2	-	-	-

H-43

Source: Pope-Reid Associates, Inc.

TABLE H-27
DEFINITION OF COSTING VARIABLES
FOR SUBCATEGORY B PLANTS FOR NONCHROME TAN

General	
Costs developed for one plant	$N_G = 1$
For Landfill	
One waste stream	$\left\{ \begin{array}{l} N_W = 1 \\ N_C = 1 \end{array} \right.$
Landfill located off site	$\left\{ \begin{array}{l} N_{ON} = 0 \\ N_{OFF} = 1 \end{array} \right.$
Waste load 13 MT/yr and originates from one stream; 4 shipments per year	$A_W = 13$ $N_M = 4$
For Disposal Impoundment	
One waste stream	$\left\{ \begin{array}{l} N_W = 1 \\ N_C = 1 \end{array} \right.$
Impoundment located on site	$\left\{ \begin{array}{l} N_{ON} = 1 \\ N_{OFF} = 1 \end{array} \right.$
1 impoundment covering 0.3 acres	$N_{SI} = 1$ $A = 0.33$
For Treatment/Storage Impoundments	
One waste stream	$\left\{ \begin{array}{l} N_W = 1 \\ N_C = 1 \end{array} \right.$
Impoundment located on site	$\left\{ \begin{array}{l} N_{ON} = 1 \\ N_{OFF} = 0 \end{array} \right.$
2 impoundments averaging 0.5 acres/impoundment	$N_{SI} = 2$ $A = 0.5$

Sources: Arthur D. Little, Inc.

TABLE H-28

**NONCHROME TAN - SMALL - SUMMARY OF COMPLIANCE COSTS -
OFF-SITE LANDFILL (FIRST RUN)**
(5)

Cost Type	Treatment/ Disposal	Compliance Activity Requirements					Contingency Plan	Closure Fund	Total
		Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training				
Capital	0	0	0	0	0		2078	0	2078
Initial	0	34	6828	1391	0		66	0	8285
Annual	876	484	0	2281	0		0	0	3650
Closure Fund	0	0	0	0	0		0	2100	2100

Source: Arthur D. Little, Inc., and Pope-Held Associates, Inc.

TABLE H-29

NONCHROME TAN - SMALL -
CALCULATION OF COMPLIANCE COSTS FOR OFF-SITE LANDFILL
(FIRST RUN)

a. Treatment/Disposal Requirements
Section 3004

Annual

Formula: TR-4 - Off-site Landfill Disposal

Calculation: $7.45 \text{ MN}_{\text{OFF}} =$
 $(7.45) (13) (1) =$
97

Formula: TR-11 - Off-site Sampling and Analysis

Calculation: $600 \text{ N}_{\text{OFF}} \text{ N}_W (1.2985) =$
779

Total Costs: Capital = 0
Initial = 0
Annual = 876
Closure Fund = 0

b. Recordkeeping and Reporting Requirements
Section 3002

Initial

Formula: RR-1 - Initial Cost of ID Code/Notification

Calculation: $\text{N}_G (\frac{1}{2} \text{ Cler} + \text{Tech}_G) =$
 $(1) (4.50 + 29) =$
34

Annual

Formula: RR-2 - Exception Reporting

Calculation: $0.05 \text{ N}_M (\frac{1}{2} \text{ Cler} + \frac{1}{2} \text{ Tech}_G) =$
 $(0.05) (4) (4.50 + 14.50) =$
4

TABLE H-29 (Continued)

Annual

Formula: RR-3 — Annual Reports

Calculation: $N_G (2 \text{ Tech}_G + \text{Cler}) =$
 $(1) [2 (29) + 9] =$
 67

Formula: RR-4 — Ongoing Review and Filing

Calculation: $110 N_{OFF} =$
 110

Formula: RR-5 — Manifest Storage and Filing

Calculation: $N_{OFF} (20 + 5 \text{ Tech}_G + 5 \text{ Supv}) =$
 $(1) (20) + 5 (29 + 27.50) =$
 303

Total Annual Cost for Section 3002

\$509.30

Total Costs: Capital = 0
 Initial = 34
 Annual = 484
 Closure Fund = 0

c. Monitoring and Testing Requirements
 Section 3001

Initial

Formula: MT-1 — Initial Cost of Applying Hazardous Waste Criteria

Calculation: $400 N_W N_C =$
 $400 (2) = 800$

Formula: MT-1a — Sampling Development Costs
 $3014 (2) = 6028$

Total Costs: Capital = 0
 Initial = 6,828
 Annual = 0
 Closure Fund = 0

TABLE H-29 (Continued)

d. Administrative Requirements
Section 3001

Initial

Formula: AD-1 – Initial Cost of Documenting Inventory

Calculation: $3 N_C N_{OFF} N_W (\text{Cler} + \text{Tech}_G) =$
 $(3) (1) (1) (1) (9 + 29) =$
 114

Formula: AD-2 – Initial Cost of Applying Hazardous Waste List

Calculation: $2 N_G N_C N_W (\text{Cler} + \text{Tech}_G) =$
 $(2) (1) (1) (2) (9.00 + 29) =$
 152

Total Initial Cost for Section 3001

266

Annual

Formula: AD-3 – Annual Cost of Incremental Supervision

Calculation: $2.2 N_G N_C \times \text{Supv} =$
 $(2.2) (1) (1) (27.50) =$
 61

Formula: AD-4 – Annual Re-evaluation of Hazardous Waste List

Calculation: $N_G (\text{Cler} + \text{Tech}_G) =$
 $(1) (9 + 29) =$
 38

Total Annual Cost for Section 3001

99

Section 3002

Initial

Formula: AD-5 – Initial Cost of Designing and Implementing
Compliance Procedures

Calculation: $N_{OFF} (16 \text{ Cler} + 19 \text{ Tech}_G) =$
 $(1) [16 (9) + 19 (29)] =$
 695

Formula: AD-6 – Initial Cost of Supervision

Calculation: $12 N_G N_{OFF} \text{Supv} =$
 $(12) (1) (27.50) =$
 330

Total Initial Costs for Section 3002

1,025

TABLE H-29 (Continued)

Annual

Formula: AD-7 — Annual Cost of Preparing Manifest

Calculation: $\frac{1}{3} N_M \text{ Cler} =$
 $(\frac{1}{3}) (4) (9) =$
 12

Formula: AD-8 — Annual Cost of Ongoing Supervision

Calculation: $24 N_{OFF} \text{ Supv} =$
 $24 (1) (27.50) =$
 660

Total Annual Costs for Section 3002

672

Section 3003

Annual

Formula: AD-9

Calculation: $\frac{3}{4} N_M \text{ Truck} (\frac{1}{12} + \frac{15}{60}) =$
 $\frac{3}{4} (4) (\$13.50) (\frac{1}{12} + \frac{15}{60}) =$
 14

Formula: AD-10

Calculation: $\frac{1}{4} N_M \text{ Truck} (\frac{1}{6} + \frac{15}{360}) =$
 $\frac{1}{4} (4) (\$13.50) (\frac{1}{6} + \frac{15}{360}) =$
 6

Total Annual Cost for Section 3003

20

Section 3004

Initial

Formula: AD-15 — Inspection Plan

Calculation: $100 N_{OFF} =$
 100

Annual

Formula: AD-16 — Inspection Plan

Calculation: $1,500 N_{OFF} =$
 1,500

Total Costs: Capital = 0
 Initial = 1,391
 Annual = 2,351
 Closure Fund = 0

TABLE H-29 (Continued)

e. Contingency Plan Requirements
Section 3004

Capital

Formula: CP-4 — Tank Containment

Calculation: $2 N_{OFF} 1,038 =$
2,076

Initial

Formula: CP-5 — Security

Calculation: $66 N_{OFF} =$
66

Total Cost: Capital = 2,076
Initial = 66
Annual = 0
Closure Fund = 0

f. Closure Fund Requirements
Section 3004

Closure Fund

Formula: CF-4 — Tanks

Calculation: $(2 \text{ Tank} + 100) N_{OFF} =$
 $[(2) (1,000) + 100] (1) =$
2,100

Total Cost: Capital = 0
Initial = 0
Annual = 0
Closure Fund = 2,100

The subsequent-run formulas were used to calculate costs associated with the disposal impoundment and the treatment/storage impoundments. Tables H-30 and H-31 summarize the results. Tables H-32 and H-33, give details of these calculations.

The total cost for the Subcategory B plant is simply the sum of the cost elements developed for each of the three technologies (Tables H-28, H-30, and H-31), multiplied by 1.14. The latter factor corrects for the fact that the production capacity totaled over all plants in the size segment does not equal the known production of the total segment. Table H-34 presents the costs for the Subcategory B plant.

Model plant costs can simply be derived from the cost figures from Subcategories A, B, and C and the plant number information contained in Table H-26.

The cost for a model plant with no impoundments is

$$C_1 = (2)C_C + (2)C_D/4$$

where:

C_C is the cost for Subcategory C and

C_D is the cost for Subcategory D (here zero, since these plants produce no waste and hence accrue no costs).

These costs are multiplied by the number of plants in the subcategory and divided by the total number of plants contributing to the cost.

The cost for a model plant with treatment/storage impoundment only is irrelevant because there are none.

The cost for a model plant with disposal impoundments only is equal to the cost for Subcategory A, because only this plant meets that criterion.

Similarly the cost for a model plant with both types of impoundments is equal to the cost for Subcategory B.

The total cost for the industry size segment is simply the sum of the cost of each subcategory multiplied by the number of plants in that subcategory. That is:

$$C_{TOTAL} = (1)C_A + (1)C_B + (2)C_C + (2)C_D$$

Total costs for the industry segment are the sums of comparable costs associated with each size segment. The annualized cost is computed from the capital, initial, annual, and closure fund costs as described in the preceding example.

TABLE H-30

**NONCHROME TAN - SMALL - SUMMARY OF COMPLIANCE COSTS -
DISPOSAL IMPOUNDMENTS (SUBSEQUENT RUN)**
(\$)

Cost Type	Treatment/ Disposal	Compliance Activity Requirements					Contingency Plan	Closure Fund	Total
		Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training				
Capital	0	0	23,000	0	0		11,222	0	34,222
Initial	0	464	3,080	456	0		0	0	4,000
Annual	0	948	4,620	145	0		0	0	5,713
Closure Fund	0	0	0	0	0		0	142,282	142,282

Sources: Arthur D. Little, Inc., and Pope-Field Associates, Inc.

TABLE H-31

**NONCHROME TAN - SMALL - SUMMARY OF COMPLIANCE COSTS -
TREATMENT/STORAGE IMPOUNDMENTS (SUBSEQUENT RUN)**

(\$)

Cost Type	Treatment/ Disposal	Compliance Activity Requirements					Contingency Plan	Closure Fund	Total
		Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training				
Capital	0	0	23,000	0	0	0	18,409	0	39,409
Initial	0	484	3,080	456	0	0	0	0	4,000
Annual	0	968	4,620	145	0	0	0	0	5,733
Closure Fund	0	0	0	0	0	0	0	308,474	308,474

Source: Arthur D. Little, Inc., and Pope-Reid Associates, Inc.

TABLE H-32

**NONCHROME TAN – SMALL –
CALCULATION OF COMPLIANCE COSTS FOR SECOND WASTE STREAM –
ON-SITE DISPOSAL IMPOUNDMENT
(SUBSEQUENT RUN)**

**a. Recordkeeping and Reporting Requirements
Section 3002**

Annual

Formula: I-RR-2 – Annual Reports

Calculation: $N_G \times (2 \text{ Tech}_G + \text{Cler}) =$
 $(1) [2 (29) + 9] =$
 67

Section 3004

Annual

Formula: I-RR-3 – Annual Reporting

Calculation: $N_{ON} (\text{Cler} + \text{Tech}_G) =$
 $1 \times (9.00 + 29.00) =$
 38

Formula: I-RR-4 – Disposal Records

Calculation: $N_{ON} (12 \text{ Tech}_G + 24 \text{ Cler} + 20) =$
 $(1) [(12) (29) + (24) (9) + 20] =$
 584

Formula: I-RR-5 – Non-incinerator, Ground-Water Monitoring

Calculation: $N_{ON} (8 \text{ Tech}_G + 3 \text{ Cler}) =$
 $(1) [(8) (29) + (3) (9)] =$
 259

Total Annual Costs for Section 3004

881

Section 3005

Initial

Formula: I-RR-7 – Subpart A

Calculation: $N_{ON} (5 \text{ Supv} + 10 \text{ Tech}_G + 4 \text{ Cler}) =$
 $(1) [(5) (27.50) + (10) (29) + (4) (9)] =$
 464

Total Costs: Capital = 0
 Initial = 464
 Annual = 948
 Closure Fund = 0

TABLE H-32 (Continued)

b. Monitoring and Testing Requirements
Section 3004

Capital

Formula: I-MT-1 – Monitoring Wells

Calculation: $23,000 N_{ON} =$
23,000

Initial

Formula: I-MT-2 – Monitoring

Calculation: $3,080 N_{ON} =$
3,080

Annual

Formula: I-MT-3 – On-site Annual Waste Analysis

Calculation: $3,400 N_{ON} =$
3,400

Formula: I-MT-5 – Monitoring

Calculation: $1,220 N_{ON} =$
1,220

Total Annual Cost for Section 3004

4,620

Total Costs:	Capital	= 23,000
	Initial	= 3,080
	Annual	= 4,620
	Closure Fund	= 0

c. Administrative Requirements
Section 3004

Annual

Formula: I-AD-4 – Weekly Inspection

Calculation: $5 N_{ON} Tech_G =$
 $(5) (1) (29) =$
145

TABLE H-32 (Continued)

Section 3005

Initial

Formula: I-AD-5

Calculation: $N_{ON} (10 \text{ Supv} + 5 \text{ Tech}_G + 4 \text{ Cler}) =$
 $(1) [(10) (27.50) + (5) (29) + (4) (9)] =$
 456

Total Costs: Capital = 0
 Initial = 456
 Annual = 145
 Closure Fund = 0

d. Contingency Plan Requirements
 Section 3004

Capital

Formula: I-CP-1 — Contingency Plan

Calculation: $N_{ON} 7,500 =$
 $(1) 7,500 =$
 7,500

Formula: I-CP-2 — Fencing

Calculation: $(325 + 48 \sqrt{43,560 \times 1.37 (A)^{0.9}}) N_{ON} (N_1) (0.5)$
 $[325 + 48 \sqrt{43,560 \times 1.37 (0.33)^{0.9}}] (1) (1) (0.5)$
 3,723

Total Capital Cost for Section 3004

11,222

Total Costs: Capital = 11,222
 Initial = 0
 Annual = 0
 Closure Fund = 0

e. Closure Fund Requirements
 Section 3004

Closure Fund

Formula: I-CF-1 — Administrative Overhead

Calculation: $500 N_{ON} N_1$
 $(500)(1)(1)$
 500

TABLE H-32 (Continued)

Formula: I-CF-8 — Nontechnical Closure Costs

Calculation: $N_{ON} N_1 \times 2,175 =$
 $1 \times 1 \times 2,175 =$
 2,175

Formula: I-CF-9 — Post-Closure Cost

Calculation: $N_{ON} N_1 \times 115,500 =$
 $(1) (1) (115,500) =$
 115,500

Formula: I-CF-12 — Closure Cost

Calculation: $[45,360 (A)^{0.982} + 25,250 (A)^{0.947}] N_{ON} N_{Si} =$
 $(1) (1) [45,360 (0.33)^{0.982} + 25,250 (0.33)^{0.947}] =$
 24,107

Total Closure Fund for Section 3004

142,282

Total Costs:	Capital	= 0
	Initial	= 0
	Annual	= 0
	Closure Fund	= 142,282

Source: Arthur D. Little, Inc.

TABLE H-33

NONCHROME TAN - SMALL - CALCULATION OF COMPLIANCE COSTS FOR SECOND WASTE STREAM - ON-SITE TREATMENT/STORAGE IMPOUNDMENTS (SUBSEQUENT RUN)

a. Recordkeeping and Reporting Requirements Section 3002

Annual

Formula: I-RR-2 - Annual Reports

Calculation: $N_G (2 \text{ Tech}_G + \text{Cler}) =$
 $(1) [(2) (29) + 9] =$
 67

Section 3004

Annual

Formula: I-RR-3 - Annual Reporting

Calculation: $N_{ON} (\text{Cler} + \text{Tech}_G) =$
 $(1) (9 + 29) =$
 38

Formula: I-RR-4 - Disposal Records

Calculation: $N_{ON} (12 \text{ Tech}_G + 24 \text{ Cler} + 20) =$
 $(1) [(12) (29) + (24) (9) + 20] =$
 584

Formula: I-RR-5 - Non-Incinerator Groundwater Monitoring

Calculation: $N_{ON} (8 \text{ Tech}_G + 3 \text{ Cler} + 20) =$
 $(1) [(8) (29) + (3) (9) + 20] =$
 279

Total Annual Cost for Section 3004

901

Section 3005

Initial

Formula: I-RR-7 - Subpart A

Calculation: $N_{ON} (5 \text{ Supv} + 10 \text{ Tech}_G + 4 \text{ Cler}) =$
 $(1) [(5) (27.50) + (10) (29) + (4) (9)] =$
 464

Total Costs: Capital = 0
 Initial = 464
 Annual = 968
 Closure Fund = 0

TABLE H-33 (Continued)

b. Monitoring and Testing Requirements
Section 3004

Capital

Formula: I-MT-1 - Monitoring Wells

Calculation: $23,000 N_{ON} =$
23,000

Initial

Formula: I-MT-2 - Monitoring

Calculation: $3,080 \times N_{ON} =$
3,080

Annual

Formula: I-MT-3 - On-site Annual Waste Analysis

Calculation: $3,400 N_{ON} =$
3,400

Formula: I-MT-5 - Monitoring

Calculation: $1,220 N_{ON} =$
1,220

Total Annual Cost

4,620

Total Costs:	Capital	=	23,000
	Initial	=	3,080
	Annual	=	4,620
	Closure Fund	=	0

c. Administrative Requirements
Section 3004

Annual

Formula: I-AD-4 - Weekly Inspection

Calculation: $5 N_{ON} \times Tech_G =$
 $(5) (1) (29) =$
145

TABLE H-33 (Continued)

Section 3005

Initial

Formula: I-AD-5

Calculation: $N_{ON} (10 \text{ Supv} + 5 \text{ Tech}_G + 4 \text{ Cler}) =$
 $(1) [(10) (27.50) + (5) (29) + (4) (9)] =$
 456

Total Costs: Capital = 0
 Initial = 456
 Annual = 145
 Closure Fund = 0

d. Contingency Plan Requirements
 Section 3004

Capital

Formula: I-CP-1 — Contingency Plan

Calculation: $N_{ON} 7,500 =$
 $(1) (7,500) =$
 7,500

Formula: I-CP-4 — Fencing

Calculation: $N_{ON} N_{Si} [325 + 48 \sqrt{43,560 \times 1.37 \times (A)^{0.9}}] (0.5) =$
 $(1) (2) [325 + 48 \sqrt{43,560 \times 1.37 \times (0.5)^{0.9}}] (0.5) =$
 8,909

Total Capital Costs for Section 3004

16,409

Total Costs: Capital = 25,318
 Initial = 0
 Annual = 0
 Closure Fund = 0

e. Closure Fund Requirements
 Section 3004

Closure Fund

Formula: I-CF-1 — Administrative Overhead

Calculation: $N_{Si} N_{ON} 500 =$
 $(2) (1) (500) =$
 1,000

TABLE H-33 (Continued)

Formula: I-CF-8 — Nontechnical Closure Cost

Calculation: $N_{Si} N_{ON} 2,175 =$
 $(2) (1) (2,175) =$
 4,350

Formula: I-CF-9 — Post-Closure Costs

Calculation: $N_{Si} N_{ON} 115,500 =$
 $(2) (1) (115,500) =$
 231,000

Formula: I-CF-12 — Closure Cost

Calculation: $N_{ON} N_{Si} [45,360 (A)^{0.982} + 25,250 (A)^{0.947}] =$
 $(1) (2) [45,360 (0.5)^{0.982} + 25,250 (0.5)^{0.947}] =$
 72,124

Total Closure Fund for Section 3004

308,474

Total Costs: Capital = 0
 Initial = 0
 Annual = 0
 Closure Fund = 308,474

TABLE H-34
NONCHROME TAN - SMALL - SUBCATEGORY B -
SUMMARY OF COMPLIANCE COSTS
(\$)

Cost Type	Treatment/ Disposal	Compliance Activity Requirements					Contingency Plan	Closure Fund	Total
		Recordkeeping & Reporting	Monitoring & Testing	Administrative	Training				
Capital	0	0	52,440	0	0		33,866	0	86,306
Initial	0	1,097	14,808	2,625	0		75	0	18,603
Annual	908	2,736	10,534	2,942	0		0	0	17,210
Closure Fund	0	0	0	0	0		0	516,256	516,256

Sources: Arthur D. Little, Inc., and Pope-Rold Associates, Inc.

APPENDIX I
COMPUTER PROGRAM

APPENDIX I

COMPUTER PROGRAM

1. UNIT COST FORMULAS

The unit cost formulas described in detail in Appendix B are summarized here. There are two sets of formulas, as described in Chapter V. The first-run costs include nontechnology-specific costs as well as technology-specific costs; these costs are used for the first disposal technology costed for each model plant subcategory. The subsequent-run costs include only technology-specific costs and are used for each additional technology utilized by the model plant subcategory. Table I-1 defines the symbols used. The first run and subsequent run formulas are given in Tables I-2 and I-3, respectively.

2. COMPUTER PROGRAM

The remainder of this appendix contains the computer program used in this analysis.

TABLE I-1

COMPLIANCE COST FORMULA NOMENCLATURE

A	Surface impoundment area (acres)
A_W	Average quantity of hazardous waste sent off site per waste stream by a generator (MT/yr) = M/N_W
Cler	Unit wage rate for a clerical worker (\$9.00/hr)
Cont	Cost of installing tank containment (\$)
M	Total hazardous waste generated annually by a model plant (MT/Yr)
N_C	Number of different monitoring and testing codes for a model plant
N_G	Number of generators represented by a model plant
N_I	Number of impoundments
N_M	Number of manifests prepared by a model plant
N_{OFF}	Number of generators that dispose of hazardous wastes in off-site facilities
N_{ON}	Number of generators that dispose of hazardous wastes in on-site facilities
N_T	Number of transporters
N_W	Number of hazardous waste streams produced by a model plant
P	Percent of plants at Level 1
Supv	Unit wage rate for a supervisory engineer or research executive (\$27.50/hr)
Tank	Cost of decontaminating and decommissioning hazardous waste storage tank (\$)
Tech _G	Unit wage rate for an engineer employed by a hazardous waste generator (\$29.00/hr)
Tech _{TSDf}	Unit wage rate for an engineer employed by a TSDf (\$35.00/hr)
Truck	Unit wage rate for a truck driver (\$13.50/hr)

Sources: Arthur D. Little, Inc.

TABLE 1-2
ISS COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS
TREATMENT/DISPOSAL REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site land treatment specific cost	TR-1	$(13,100 + 27 \sqrt{60M}) N_{ON}$	Capital	3004-§265.272	Land treatment runoff control
On-site landfill specific cost	TR-2	$(13,100 + 27 \sqrt{127M^{(0.481)}} N_{ON}$	Annual	3004-§265.272	Landfill runoff control
On-site landfill specific cost	TR-3	$(87(M)^{0.576} + 141.5(M)^{0.597}) N_{ON}$	Annual	3004-§265.314	Annual closure of a cell
Off-site sanitary landfill specific cost	TR-4	7.45 MN _{OFF}	Annual	3004-§265.314	Landfill treatment/disposal cost
Off-site incinerator (halogenated) specific cost	TR-5	18.32 MN _{OFF}	Annual	3004-§265.343	Treatment/disposal cost associated with incinerators equipped with scrubbers
Off-site incinerator (hydrocarbon) specific cost	TR-6	5.74 MN _{OFF}	Annual	3004-§265.343	Treatment/disposal cost associated with incinerators not equipped with scrubbers
Off-site secure landfill specific cost	TR-7	9.33 MN _{OFF}	Annual	3004-§265.343	Off-site treatment/disposal cost
Off-site secure drummed waste specific cost	TR-8	9.33 MN _{OFF}	Annual	3004-§265.343	Off-site treatment/disposal cost
Off-site dump specific cost	TR-9	6.74 MN _{OFF}	Annual	3004-§265.343	Off-site disposal cost
Off-site land treatment specific cost	TR-10	8.79 MN _{OFF}	Annual	3004-§265.343	Off-site disposal cost
Off-site specific cost for generators with average off-site wastes (M/N _W) ≤ 80 tons	TR-11	600 N _{OFF} N _W (1.2686)	Annual	3004-§264.13	Sampling and analysis of wastes transported off site
Off-site specific cost for generators with average off-site wastes 80 tons < (M/N _W) ≤ 160 tons	TR-12	N _{OFF} N _W A _W (800/160 + 50/20) (1.2686)	Annual	3004-§264.13	Sampling and analysis of wastes transported off site
Off-site specific cost for generators with average off-site wastes (M/N _W) > 160 tons	TR-13	N _{OFF} N _W (A _W (50/20) + 800) (1.2686)	Annual	3004-§264.13	Sampling and analysis of wastes transported off site

TABLE 1-2 (Continued)

ISS COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS
RECORDKEEPING AND REPORTING REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCHA Section	Comments
Cost to all generators	RR-1	$N_G (1/2 \text{ Cler} + \text{Tech}_G)$	Initial	3002-\$262.12	Initial cost of ID code/notification
Off-site specific cost	RR-2	$0.06 N_M (1/2 \text{ Cler} + 1/2 \text{ Tech}_G)$	Annual	3002-\$262.42	Exception reporting
Cost to all generators	RR-3	$N_G (2 \text{ Tech}_G + \text{Cler})$	Annual	3002-\$262.41	Annual reports
Off-site specific cost	RR-4	$110 N_{OFF}$	Annual	3002-\$262.40	Recordkeeping, ongoing review and filing
Off-site specific cost	RR-5	$N_{OFF} (20 + 5 \text{ Tech}_G + 5 \text{ Supv})$	Annual	3002-\$262.23	Manifest storage and filing
On-site specific cost	RR-6	$N_{ON} (20 + 12 \text{ Cler})$	Annual	3004-\$264.16	Training recordkeeping
On-site specific cost	RR-7	$N_{ON} (\text{Cler} + \text{Tech}_G)$	Annual	3004-\$264.75	Annual reporting
On-site specific cost	RR-8	$N_{ON} (24 \text{ Tech}_G + 72 \text{ Cler} + 20)$	Annual	3004-\$264.73	Disposal records
On-site specific cost (non-incinerator)	RR-9	$N_{ON} (8 \text{ Tech}_G + 3 \text{ Cler} + 20)$	Annual	3004-\$265.94	Non-incinerator groundwater monitoring
On-site specific cost (incinerator only)	RR-10	$2 N_{ON} (8 \text{ Tech}_G + 2 \text{ Supv} + 3 \text{ Cler} + 20)$	Annual	3004-\$265.94	Incinerator groundwater monitoring
On-site specific cost	RR-11	$N_{ON} (\text{Tech}_G + 1/2 \text{ Cler})$	Initial	3005-\$122.24 3005-\$122.25	Initial cost of ID code application/notification
On-site specific cost	RR-12	$N_{ON} (20 \text{ Supv} + 40 \text{ Tech}_G + 15 \text{ Cler})$	Initial	3005-Subpart A	Preparation of Part A of permit

TABLE 1-2 (Continued)

**ISS COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS
MONITORING AND TESTING REQUIREMENTS**

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
Waste stream specific cost	MT-1	Monitoring and Testing Code	Initial	3001-Subparts B & C	Initial cost of applying hazardous waste criteria
		A			
		B			
		C			
		NA			
		Cost			
		45N _W N _G			
		500N _W N _G			
		400N _W N _G			
		400N _W N _G			
Waste stream specific cost	MT-1a	3,014 N _G N _W	Initial	3004-§ 265.13	Sampling development
On-site, disposal mode-specific cost (non-incinerator only)	MT-2	23,000 N _{ON}	Capital	3004-§ 265.91	Monitoring wells
On-site, disposal mode-specific cost (secure and sanitary landfill, impoundment, dump, and drummed wastes only)	MT-3	3,080 N _{ON}	Initial	3004-§ 265.91	Monitoring
On-site disposal mode-specific cost (secure or sanitary landfill, impoundment, dump, or drummed waste disposal only)	MT-4	3,400 N _{ON}	Annual	3004-§ 265.13	Annual on-site waste analysis cost for landfills and disposal impoundments
On-site disposal mode-specific cost (incinerators only)	MT-5	11,000 N _{ON}	Annual	3004-§ 265.13	Annual on-site waste analysis cost for incinerators
On-site, disposal mode-specific cost (secure and sanitary landfill, impoundment, dump, and drummed wastes only)	MT-6	1,220 N _{ON}	Annual	3004-§ 265.91	Monitoring
On-site land treatment specific cost	MT-6a	3,300 N _{ON}	Initial	3004-§ 265.278	Initial testing
On-site, land treatment specific cost	MT-7	39,900 N _{ON}	Annual	3004-§ 265.278	Unsatuated zone monitoring

TABLE 1-2 (Continued)

ISS COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS

ADMINISTRATIVE REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
Off-site, disposal mode/waste stream specific cost	AD-1	$3N_{OFF}N_W(\text{Cler} + \text{Tech}_G)$	Initial	3001-Subparts A, B, C, D	Initial cost of documenting inventory
Waste stream/disposal mode specific cost	AD-2	$2N_GN_W(\text{Cler} + \text{Tech}_G)$	Initial	3001-Subparts A, B, C, D	Initial cost of applying hazardous waste list
Waste stream/disposal mode specific cost	AD-3	$2.2N_GN_W \text{ Supv}$	Initial	3001-Subparts A, B, C, D	Initial cost of incremental supervision
Cost to all generators	AD-4	$N_G(\text{Cler} + \text{Tech}_G)$	Annual	3001-Subparts A, B, C, D	Annual re-evaluation of hazardous waste list
Off-site specific cost	AD-5	$N_{OFF}(116 \text{ Cler} + 19 \text{ Tech}_G)$	Initial	3002-§§262.11, 262.12	Initial cost of designing and implementing compliance procedures
Site-specific cost	AD-6	$12N_{OFF} \text{ Supv}$	Initial	3002-§262.11	Initial cost of supervision
Manifest specific cost	AD-7	$1/3 N_M \text{ Cler}$	Annual	3002-§§262.20, 262.21, 262.22, 262.23	Annual cost of preparing manifest
Off-site specific cost	AD-8	$24N_{OFF} \text{ Supv}$	Annual	3002-§§262.30, 262.31, 262.32	Annual cost of ongoing supervision
Disposal mode/manifest specific cost	AD-9	$3/4 N_M \text{ Truck} [(1/2)(116/60)(0.16)]$	Annual	262.33, 262.34	Transportation
Disposal mode/manifest specific cost	AD-10	$1/4 N_M \text{ Truck} [(116 + 15/60)(0.16)]$	Annual	3003-§263.20	Transportation
On-site specific cost	AD-11	$N_{ON}(40 \text{ Tech}_G + 30 \text{ Cler})$	Initial	3003-§263.20	Financial requirement plan design
On-site specific cost	AD-12	$N_{ON}(80 \text{ Tech}_G + 32 \text{ Cler})$	Initial	3004-§265.142	Waste plan
On-site specific cost	AD-13	$N_{ON}(36 \text{ Tech}_G + 15 \text{ Cler})$	Initial	3004-§264.13	Recordkeeping, logging, incidents
On-site specific cost	AD-14	$15N_{ON} \text{ Supv}$	Initial	3004-§285.70	General overhead
Off-site specific cost	AD-15	$100N_{OFF}$	Initial	3004-§285.70	Inspection plan
Off-site specific cost	AD-16	$1600N_{OFF}$	Annual	3004-§264.15	Inspection
On-site, disposal mode specific cost	AD-17	$25N_{ON} \text{ Tech}_G$	Annual	3004-§264.15	Weekly inspections
On-site, disposal mode specific cost	AD-18	$N_{ON}(40 \text{ Supv} + 20 \text{ Tech}_G + 15 \text{ Cler})$	Initial	3005-§122.24	Preparation of permit application

TABLE 1-2 (Continued)

ISS COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS

TRAINING REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site specific cost	TG-1	10,175 N_{OH}	Initial	3004-§264.16	Development of training program
On-site specific cost for landfills and incinerators with $\leq 20,000$ MT/yr waste or impoundments and land treatment with ≤ 2.15 acres	TG-2	3,000 N_{OH}	Annual	3004-§264.16	On-the-job or classroom training of 10 employees
On-site specific costs for landfills and incinerators with $> 20,000$ MT/yr waste or land treatment and impoundments with > 2.15 acres	TG-3	4,500 N_{OH}	Annual	3004-§264.16	On-the-job or classroom training of 15 employees

TABLE 1-2 (Continued)

IES COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS
CONTINGENCY PLAN REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments										
On-site specific cost	CP-1	$10,000 N_{ON}$	Capital	3004-\$264.51	Contingency plan										
On-site land treatment specific cost	CP-2	$(1325 + 48 \sqrt{60 M}) N_{ON}$	Capital	3004-\$264.51	Fencing										
On-site impoundment specific cost	CP-3	$(1325 + 48 \sqrt{43,560 \times 1.37 (A)^{0.5}}) N_{ON} N_1^{(0.5)}$	Capital	3004-\$265.51	Fencing										
Off-site specific cost	CP-4	<table><tr><th>M</th><th>Cont</th></tr><tr><td>0-2,000</td><td>1,038</td></tr><tr><td>2,001-5,500</td><td>1,786</td></tr><tr><td>5,501-20,000</td><td>2,309</td></tr><tr><td>>20,000</td><td>3,559</td></tr></table>	M	Cont	0-2,000	1,038	2,001-5,500	1,786	5,501-20,000	2,309	>20,000	3,559	Capital	3004-\$265.192	Tank containment
M	Cont														
0-2,000	1,038														
2,001-5,500	1,786														
5,501-20,000	2,309														
>20,000	3,559														
Off-site specific cost	CP-5	$68 N_{OFF}$	Initial	3004-\$265.194	Security										
On-site landfill specific cost	CP-6	$(1325 + 48 \sqrt{127 (M)^{0.511}}) N_{ON}$	Annual	3004-\$264.51	Fencing										
On-site specific cost	CP-7	$N_{ON} (30 \text{ Supv} + 16 \text{ Tech}_G + 10 \text{ Cler})$	Initial	3005-\$122.24	Description of security procedures										

TABLE I-3 (Continued)

ISS COMPLIANCE COSTING FORMULAS - FIRST RUN COSTS
CLOSURE FUND REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site specific cost; off-site tanks - specific cost	CF-1	500 N 500 N _{OFF}	Closure fund	3004-\$265.111, 265.112, \$265.113, 265.117 \$265.118	Administrative overhead
On-site sanitary or secure landfill, dump or drummed waste specific cost	CF-2	M Range 0- 1,000 1,001- 2,000 2,001- 5,500 5,501-20,000 > 20,000	Closure fund	3004-\$265.310	Closure costs
On-site secure or sanitary landfill, dump or drummed waste specific cost	CF-3	M Range 0- 1,000 1,001- 2,000 2,001- 5,500 5,501-20,000 > 20,000	Closure fund	3004-\$265.310	Post-closure costs
Off-site specific cost	CF-4	(2 Tank + 100) N _{OFF} M Tank 0-2,000 1,000 2,001-5,500 1,800 5,501-20,000 2,200 > 20,000 4,400	Closure fund	3004-\$265.197	Tank
On-site land treatment specific cost	CF-5	Closure costs = 4,245 N _{ON} Post closure costs = 0	Closure fund	3004-\$265.280	Post-closure and closure costs

TABLE 1-2 (Continued)
ISS COMPLIANCE COSTING FORMULAS - FIRST RIIN COSTS
CLOSURE FUND REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site incinerator (halogenated) specific cost	CF-6	M Range 0-1,000 1,001-5,000 5,001-10,000 10,001-20,000 >20,000 Total Closure Costs 223,000 N_{ON} 200,000 N_{ON} 338,000 N_{ON} 311,000 N_{ON} 320,000 N_{ON}	Closure fund	3004-§265.351	Total closure costs for incinerators equipped with scrubbers
On-site incinerator (hydrocarbon) specific cost	CF-7	M Range 0-1,000 1,001-5,000 5,001-10,000 10,001-20,000 >20,000 Total Closure Costs 198,000 N_{ON} 202,000 N_{ON} 204,000 N_{ON} 184,000 N_{ON} 195,000 N_{ON}	Closure fund	3004-§265.351	Total closure costs for incinerators not equipped with scrubbers
On-site incinerator specific cost	CF-8	M Range 0-1,000 1,001-5,000 5,001-10,000 10,001-20,000 >20,000 Closure Costs 1,600 N_{ON} 1,820 N_{ON} 2,750 N_{ON} 3,780 N_{ON} 6,750 N_{ON}	Closure fund	3004-§265.351	Total incinerator post-closure costs
On-site disposal and treatment/storage impoundment specific cost	CF-9	Impoundment Size (acre) 0-1 1-2.15 >2.15 Closure Costs 2,175 N_I 2,405 N_I 2,750 N_I	Closure fund	3004-§265.238	Impoundment closure costs
On-site disposal impoundment specific cost	CF-10	Impoundment Size (acre) 0-1 1-2.15 >2.15 Post-Closure Costs 115,500 N_I 124,500 N_I 131,100 N_I	Closure fund	3004-§265.238	Post-closure costs
On-site landfill specific cost	CF-11	$87 (M)^{0.575} + 141.5 (M)^{0.597} N_{ON}$	Closure fund	3004-§265.310	Landfill size-dependent closure cost
On-site land treatment specific cost	CF-12	$10.375 N_{ON}$	Closure fund	3004-§265.280	Land treatment size-dependent closure cost
On-site impoundment specific cost	CF-13	$N_I (45,350 (A)^{0.982} + 25,250 (A)^{0.947}) N_{ON}$	Closure fund	3004-§265.228	Impoundment size-dependent closure cost

Source: Arthur D. Little, Inc., estimates.

TABLE I-3

ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN COSTS DISPOSAL REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site land treatment specific cost	I-TR-1	$(13,100 + 27 \sqrt{60M}) N_{ON}$	Capital	3004-§265.272	Land treatment runoff control
On-site landfill specific cost	I-TR-2	$(13,100 + 27 \sqrt{127M^{(0.5511)}}) N_{ON}$	Annual	3004-§265.272	Landfill runoff control
On-site landfill specific cost	I-TR-3	$(87 (M)^{0.575} + 141.5 (M)^{0.597}) N_{ON}$	Annual	3004-§265.314	Annual closure of a cell
Off-site sanitary landfill specific cost	I-TR-4	$7.46 MN_{OFF}$	Annual	3004-§265.314	Landfill disposal cost
Off-site incinerator (halogenated) specific cost	I-TR-5	$18.32 MN_{OFF}$	Annual	3004-§265.343	Disposal cost associated with incinerators equipped with scrubbers
Off-site incinerator (hydrocarbon) specific cost	I-TR-6	$5.74 MN_{OFF}$	Annual	3004-§265.343	Disposal cost associated with incinerators not equipped with scrubbers
Off-site secure landfill specific cost	I-TR-7	$9.33 MN_{OFF}$	Annual	3004-§265.343	Off-site disposal cost
Off-site secure drummed waste specific cost	I-TR-8	$9.33 MN_{OFF}$	Annual	3004-§265.343	Off-site disposal cost
Off-site dump specific cost	I-TR-9	$5.74 MN_{OFF}$	Annual	3004-§265.343	Off-site disposal cost
Off-site, land treatment specific cost	I-TR-10	$8.76 MN_{OFF}$	Annual	3004-§265.343	Off-site disposal cost
Off-site specific cost for generators with average off-site wastes (M/Nw) ≤ 80 tons	I-TR-11	$600 N_{OFF} N_W (1.2986)$	Annual	3004-§264.13	Sampling and analysis of wastes transported off-site
Off-site specific costs for generators with average off-site wastes 80 tons < (M/Nw) ≤ 160 tons	I-TR-12	$N_{OFF} N_W A_W (800/160 + 50/20) (1.2986)$	Annual	3004-§264.13	Sampling and analysis of wastes transported off-site
Off-site specific costs for generators with average off-site wastes (M/Nw) > 160 tons	I-TR-13	$N_{OFF} N_W (A_W 50/20 + 800) (1.2986)$	Annual	3004-§264.13	Sampling and analysis of wastes transported off-site

TABLE I-3 (Continued)

**ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN COSTS
RECORDKEEPING AND REPORTING REQUIREMENTS**

<u>Formula Applicability</u>	<u>Formula Number</u>	<u>Formula</u>	<u>Cost Type</u>	<u>RCRA Section</u>	<u>Comments</u>
Off-site specific cost	I-RR-1	$0.05 N_M (1/2 \text{ Clar} + 1/2 \text{ Tech}_G)$	Annual	3002-§262.42	Exception reporting
Cost to all generators	I-RR-2	$N_G (2 \text{ Tech}_G + \text{Clar})$	Annual	3002-§262.41	Annual reporting
On-site specific cost	I-RR-3	$N_{ON} (\text{Clar} + \text{Tech}_G)$	Annual	3004-§264.75	Annual reporting
On-site specific cost	I-RR-4	$N_{ON} (12 \text{ Tech}_G + 24 \text{ Clar} + 20)$	Annual	3004-§264.73	Disposal records
On-site specific cost (non-incineration)	I-RR-5	$N_{ON} (8 \text{ Tech}_G + 3 \text{ Clar} + 20)$	Annual	3004-§265.94	Non-incinerator, groundwater monitoring
On-site specific cost (incineration only)	I-RR-6	$2 N_{ON} (8 \text{ Tech}_G + 2 \text{ Supv} + 3 \text{ Clar} + 20)$	Annual	3004-§265.94	Groundwater monitoring of incinerated disposal waste
On-site specific cost	I-RR-7	$N_{ON} (5 \text{ Supv} + 10 \text{ Tech}_G + 4 \text{ Clar})$	Initial	3005-Subpart A	Preparation of Part A of permit

TABLE I-3 (Continued)
ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN COSTS
MONITORING AND TESTING REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site specific cost (non-incinerator only)	I-MT-1	$23,000 N_{ON}$	Capital	3004-§285.91	Monitoring wells
Waste stream specific cost (except treatment/storage impoundments)	I-MT-1a	$3,010 N_W N_{ON}$	Initial	3004-§285.13	Sampling development
On-site, disposal mode-specific cost (secure and sanitary landfill, impoundment, dump and drummed waste only)	I-MT-2	$3,080 N_{ON}$	Initial	3004-§285.13	Monitoring
On-site specific costs (sanitary or secure landfill, drummed waste, dump, disposal impoundment only)	I-MT-3	$3,400 N_{ON}$	Annual	3004-§285.13	On-site annual waste analysis
On-site specific cost (incinerator only)	I-MT-4	$11,000 N_{ON}$	Annual	3004-§285.13	On-site annual waste analysis
On-site disposal mode-specific cost (secure and sanitary landfill, impoundment, dump and drummed waste only)	I-MT-5	$1,220 N_{ON}$	Annual	3004-§285.13	Monitoring
On-site land treatment specific cost	I-MT-6	$3,300 N_{ON}$	Initial	3004-§285.270	Initial testing
On-site land treatment specific cost	I-MT-7	$39,800 N_{ON}$	Annual	3004-§285.278	Unsaturated zone monitoring

TABLE I-3 (Continued)

ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN COSTS
ADMINISTRATIVE REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
Manifest specific cost	I-AD-1	$\frac{1}{3} N_M \text{ Cler}$	Annual	3002 - §§262.20, 262.21, 262.22, 262.23	Manifest preparation
Off-site specific cost	I-AD-2	$\frac{3}{4} N_M \text{ Truck} \times \{ \frac{1}{12} + (15/60)(0.15) \}$	Annual	3003 - §263.20	Manifest signature
Off-site specific cost	I-AD-3	$\frac{1}{4} N_M \text{ Truck} \times \{ \frac{1}{6} + (15/60)(0.15) \}$	Annual	3003 - §263.20	Manifest signature
On-site specific cost	I-AD-4	$5 N_{ON} \text{ Tech}_G$	Annual	3004 - §264.15	Weekly inspection
On-site specific cost	I-AD-5	$N_{ON} (10 \text{ Supv} + 5 \text{ Tech}_G + 4 \text{ Cler})$	Initial	3005 - §122.24	Supervision of permit application

TABLE I-3 (Continued)

ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN COSTS
CONTINGENCY PLAN REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
Off-site specific costs	I-CP-5	2N _{OFF} CONT	Capital	3004 - §265.192	Tank Containment
		M Range			
		CONT			
		0- 2,000			
		2,001- 5,500			
On-site impoundment specific costs	I-CP-2	5,501-20,000	Capital	3004 - §265.51	Fencing
		>20,000			
		CONT			
		1,038			
		1,786			
On-site land treatment specific costs	I-CP-3	(325 + 48 $\sqrt{43,560 \times 1.37 (A)^{0.2} N_{ON} N_1 (0.5)}$)	Capital	3004 - §264.51	Fencing
		60M) N _{ON}			
		CONT			
		1,038			
		1,786			
On-site landfill specific costs	I-CP-4	(325 + 48 $\sqrt{127 (M)^{0.45} N_{ON}}$)	Annual	3004 - §264.51	Fencing
		CONT			
		1,038			
		1,786			
		2,308			
		3,559			

Source: Arthur D. Little, Inc.

TABLE I-3 (Continued)

ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN
CLOSURE FUND REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comments
On-site sanitary or secure landfill, dump or drummed waste specific cost	I-CF-2	M Range	Closure fund	3004 - §205.310	Non-technical closure cost
		0- 1,000			
		1,001- 2,000			
		2,001- 5,500			
		5,501-20,000			
On-site secure or sanitary landfill, dump or drummed waste specific cost	I-CF-3	M Range	Closure fund	3004 - §205.310	Non-technical post-closure cost
		0- 1,000			
		1,001- 2,000			
		2,001- 5,500			
		5,501-20,000			
On-site land treatment specific cost	I-CF-4	M Range	Closure fund	3004 - §205.280	Post-closure and closure cost
		0- 1,000			
		1,001- 2,000			
		2,001- 5,500			
		5,501-20,000			
On-site incinerator (halogenated) specific cost	I-CF-5	M Range	Closure fund	3004 - §205.351	Total closure cost for incinerators equipped with scrubbers
		0- 1,000			
		1,001- 5,000			
		5,001-10,000			
		10,001-20,000			
On-site incinerator (hydrocarbon) specific cost	I-CF-6	M Range	Closure fund	3004 - §205.351	Total closure cost for incinerators not equipped with scrubbers
		0- 1,000			
		1,001- 5,000			
		5,001-10,000			
		10,001-20,000			

TABLE I-3 (Continued)
ISS COMPLIANCE COSTING FORMULAS - SUBSEQUENT RUN
CLOSURE FUND REQUIREMENTS

Formula Applicability	Formula Number	Formula	Cost Type	RCRA Section	Comment
On-site incinerator specific cost	I-CF-7	M Range	Closure fund	3004-\$265.361	Total incinerator post-closure cost
		0-1,000			
		1,001-5,000			
		5,001-10,000			
		10,001-20,000			
On-site impoundment specific cost	I-CF-8	Impoundment Size (acre)	Closure fund	304-\$265.238	Impoundment closure cost
		0-1			
		1-2.15			
		>2.15			
		Impoundment Size (acre)			
On-site impoundment specific cost	I-CF-9	Impoundment Size (acre)	Closure fund	3004-\$265.238	Post-closure cost
		0-1			
		1-2.15			
		>2.15			
		Impoundment Size (acre)			
On-site landfill impoundment or treatment/storage impoundment specific cost	I-CF-10	Impoundment Size (acre)	Closure fund	3004-\$265.310	Landfill site-dependent closure cost
		0-1			
		1-2.15			
		>2.15			
		Impoundment Size (acre)			
On-site landfill impoundment or treatment/storage impoundment specific cost	I-CF-11	Impoundment Size (acre)	Closure fund	3004-\$265.280	Land treatment site-dependent closure cost
		0-1			
		1-2.15			
		>2.15			
		Impoundment Size (acre)			
On-site landfill impoundment or treatment/storage impoundment specific cost	I-CF-12	Impoundment Size (acre)	Closure fund	3004-\$265.228	Impoundment site-dependent closure cost
		0-1			
		1-2.15			
		>2.15			
		Impoundment Size (acre)			
Off-site specific cost	I-CF-13	Impoundment Size (acre)	Closure Fund	3004-\$265.197	Tank Closure
		0-1			
		1-2.15			
		>2.15			
		Impoundment Size (acre)			

Sources: Arthur D. Little, Inc.

COMPUTER PROGRAM

I-17(a)

```

/* .....
/* Written in 1980 at:  Arthur D. Little, Inc.
/*                      Cambridge, Mass. 02140
/* .....
/* .....
/* .....

annualize_npv_: procedure
    (first_year, annual, capital, closure_amt) returns (float dec (6));

dc1 first_year float dec (6),
    annual float dec (6),
    capital float dec (6),
    npv float dec (6),
    closure_amt float dec (6),
    clos_period float dec ext 0 atc,
    cl_array (20) float dec int ((20)0e0),
    result float dec (8),
    factor float dec,
    i fixed bin,
    disc (20) static float dec int ((5)1.85e-1, (15)1.57e-1),
    lnfl (20) static float dec int ((5)1e-1, (15)7e-2);

Xpage:
    factor = (1e0+lnfl (1))*5/ (1e0+disc (1))*5;
    do i = 1 to clos_period;
        cl_array (i) = closure_amt/clos_period;
    end;
    npv = (first_year+capital)* (1+lnfl (1)*5e-1)/ (1.0+disc (1)*5e-1);
    do i = 1 to 5;
        npv = npv + (annual+cl_array (i))* (1e0+lnfl (1))*1/ (1e0+disc (1))*1;
    end;
    do i = 6 to 20;
        npv = npv +
            (annual+cl_array (i))*factor*
            (1e0+lnfl (1))* (1-5)/ (1e0+disc (1))*1/ (1-5);
    end;
    result = npv * .1814;
    return (result);

end annualize_npv_;
/* was 171837 */

```

/* annualized net present value */

box.plt

01/29/81 1856,8rew 01/29/81 1450.2

208938

```

/* ..... */
/* Written in 1980 at:  Arthur D. Little, Inc.
/*                      Cambridge, Mass. 02140
/* ..... */
/* ..... */
/* ..... */
/* ..... */
/* Coded 7-15-80 by K. A. Speelman
/* changed 8-10-80 by KAS to use sicosts
/* ..... */

box: procedure;

/* program to distinguish si/~sl and regulated/~regulated generators */

%include rcra;

dcl totcosts (4, 2, 2) float dec,
    num_pl (4, 2, 2) float dec,
    impact (4, 2, 2) float dec,
    pctva (2, 2) float dec,
    value_add (70, 4, 2, 2) float dec ext static,
    value_added (4, 2, 2) float dec,
    vadd (2, 2) float dec,
    tot_pl (4, 2, 2) float dec,
    cost_per_pl (4, 2, 2) float dec,
    ccpp (2, 2) float dec,
    costs (4, 2, 2) float dec,
    (no_imp, disp_imp, te_imp, both_imp, disp_pp, sa_disp, te_pp, sa_te) float dec;

dcl (fve, ann, cap, ccost) float dec;
dcl (inp (4), num_plants (4)) float dec;
dcl (va (4), tva (4)) float dec;
dcl (sl_present, nstat, line_num, n) fixed bin;
dcl x (2) fixed bin init (6, 4);
dcl pageno pic"zz9" init (000);
dcl page_str char (3) var;
dcl (tr init (1), fr init (2), rr init (3), mt init (4), ad init (5), cp init (6), nfr init (8)) static fixed
    bin;
dcl (name, segment) char (60) var;
label (3, 3) char (12) var init ("plants", "", "");
"Surface", "", "";
"....." "Impoundments";
label2 (3, 2) char (7) var init ("Without", "With", "", "", "", "");
sizestr (4) char (7) var init ("Large", "Medium", "Small", "Sdg");
sic char (5) var;
dcl (box, sat, mod, i, j, alen, argno init (1), num_models static init (4)) fixed bin,
    ec fixed bin (35),
    (month, dom, year, hour, minute, second, dow) fixed bin,
    (bits, microsecond) fixed bin (71),
    code fixed bin (35);
sun fixed bin init (1);
dcl str1 char (64) var init ("SURFACE IMPOUNDMENT COSTS REPORT");
arg char (alen) unaligned based (aptr),

```

```

(str2, old_ind, industry, ind_save) char (60) var,
filler char (3) var init (" ").
zone char (3) init (" ").
shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT",
mo (12) char (10) var init ("January", "February", "March", "April", "May", "June", "July",
"August", "September", "October", "November", "December"),
days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");
do1 (more_ext, repeat) bit (1) aligned init ("1'b");
do1 (industry_sw, model_sw, no_report, tt) bit (1) aligned init ("0'b");
do1 aptr pointer;

do1 (substr, index, sum, length, char, clock, ltrim, lineno) builtin,
endpage condition,
(sysprint output, report output, datafile input) file stream,
cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35)),
header_ext entry (char (64) var, file),
annualize_npv_ext entry (float dec, float dec, float dec, float dec) returns (float dec);
do1 decode_clock_value_sdate_time entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
fixed bin, char (3), fixed bin (35));

Xpage:
on endpage (report) begin;
line_num = lineno (report);
do while (line_num < 59);
put file (report) skip;
line_num = lineno (report);
end;
page_str = ltrim (char (pageno));
put file (report) edit ("PAGE", page_str) (col (125), a (4), x (1), a);
put file (report) page;
pageno = pageno + 1;
put file (report) edit (mo (month), dom, " ", year, hour, ": ",
filler, ltrim (char (minute)), shift (sun), days (dow))
(col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
x (1), f (2), a (1), a, a, x (1), a, x (1), a (3));
put file (report) skip;
call header_ (str1, report);
put file (report) skip (2);
end;
/* begin block */

arglp:
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to proceed;
else if arg = "-industry" | arg = "-ind" then industry_sw = "1'b";
else if arg = "-norep" then no_report = "1'b";
else if arg = "-tt" then tt = "1'b";
else if arg = "-model" | arg = "-md" then model_sw = "1'b";
else do;
put skip list ("Args are: -industry");
put skip (2);
return;
end;
argno = argno + 1;
go to arglp;

proceed:
open file (report) title ("vfile_box.report") print linesize (135);
open file (datafile) title ("vfile_sla.data");

```

```

bits = clock ();
call decode_clock_value,$date_time
(bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code);
if hour = 12 then do;
  if minute = 0 then sun = 3;
  else sun = 2;
end;
else if hour = 0 then do;
  hour = 12;
  if minute = 0 then sun = 4;
end;
else if hour > 12 then do;
  hour = hour - 12;
  sun = 2;
end;
if minute < 10 then filler = '0';

do mod = 1 to 4;
  tva(mod) = 0e0;
  tnp(mod) = 0e0;
  do l = 1 to 2;
    do nstat = 1 to 2;
      tot_pl(mod, l, nstat) = 0e0;
      totcosts(mod, l, nstat) = 0e0;
    end;
  end;
end;

smt = 1;
str2 = ind_struct(smt).ind_name;
industry = substr(str2, 1, index(str2, ':')-1);
ind_save = industry;
do while (more_smt);
  va(*) = 0e0;
  num_plants(*) = 0;
  costs(*,*,*) = 0e0;
  num_pl(*,*,*) = 0;
  do while (industry = ind_save & more_smt & repeat);
    get file (datafile) list (name, sic, si_present);
    do mod = 1 to num_models;
      va(mod) = va(mod) + ind_struct(smt).value_add(mod);
      tva(mod) = tva(mod) + ind_struct(smt).value_add(mod);
      do nstat = 1 to 2;
        get file (datafile) list (no_imp, disp_imp, ts_imp, both_imp,
          disp_pp, sa_disp, ts_pp, sa_ts);
        num_pl(mod, 2, nstat) = num_pl(mod, 2, nstat) + disp_imp + ts_imp + both_imp;
        num_pl(mod, 1, nstat) = num_pl(mod, 1, nstat) + no_imp;
        num_plants(mod) = num_plants(mod) + no_imp + disp_imp + ts_imp + both_imp;
        tot_pl(mod, 1, nstat) = tot_pl(mod, 1, nstat) + no_imp;
        tot_pl(mod, 2, nstat) = tot_pl(mod, 2, nstat) + disp_imp + ts_imp + both_imp;
        do l = 1 to 2;
          fye = sicost$(smt, l, nstat, mod, l);
          ann = sicost$(smt, l, nstat, mod, 2);
          cap = sicost$(smt, l, nstat, mod, 3);
          clicost = sicost$(smt, l, nstat, mod, 4);
          costs(mod, l, nstat) =
            costs(mod, l, nstat) + annualize_npv_((fye, ann, cap, clicost);
            costs(mod, l, nstat) = costs(mod, l, nstat) + l*-6;

```



```

("Model:") (col (69), a)
(sizestr (n)) (col (76), a)
("-") (col (93), a)
("Model:") (col (95), a)
(sizestr (n+1)) (col (102), a)
("-") (col (117), a);
put file (report) skip edit ("-") (col (15), a)
("No. Plants:") (col (16), a)
(num_pl (n, box, 1)) (col (28), p"zz,zzz")
("-") (col (41), a)
("No. Plants:") (col (42), a)
(num_pl (n+1, box, 1)) (col (54), p"zz,zzz")
("-") (col (66), a)
("No. Plants:") (col (68), a)
(num_pl (n, box, 2)) (col (80), p"zz,zzz")
("-") (col (93), a)
("No. Plants:") (col (94), a)
(num_pl (n+1, box, 2)) (col (106), p"zz,zzz")
("-") (col (117), a);
put file (report) skip edit ("-") (col (15), a)
("Tot. Cost(MM$):") (col (16), a)
(costs (n, box, 1)) (col (31), p"$,$$$v,$$")
("-") (col (41), a)
("Tot. Cost(MM$):") (col (42), a)
(costs (n+1, box, 1)) (col (57), p"$,$$$v,$$")
("-") (col (66), a)
("Tot. Cost(MM$):") (col (68), a)
(costs (n, box, 2)) (col (83), p"$,$$$v,$$")
("-") (col (93), a)
("Tot. Cost(MM$):") (col (94), a)
(costs (n+1, box, 2)) (col (109), p"$,$$$v,$$")
("-") (col (117), a);
put file (report) skip edit ("-") (col (15), a)
("Val. Add(MM$):") (col (16), a)
(value_add (smt-1, n, box, 1)) (col (31), p"zz,zzzv,zz")
("-") (col (41), a)
("Val. Add(MM$):") (col (42), a)
(value_add (smt-1, n+1, box, 1)) (col (57), p"zz,zzzv,zz")
("-") (col (66), a)
("Val. Add(MM$):") (col (68), a)
(value_add (smt-1, n, box, 2)) (col (83), p"zz,zzzv,zz")
("-") (col (93), a)
("Val. Add(MM$):") (col (94), a)
(value_add (smt-1, n+1, box, 2)) (col (109), p"zz,zzzv,zz")
("-") (col (117), a);
put file (report) skip edit ("-") (col (15), a)
("Cost/Plant:") (col (16), a)
(cost_per_pl (n, box, 1)) (col (27), p"$$$,$$$,$$$")
("-") (col (41), a)
("Cost/Plant:") (col (42), a)
(cost_per_pl (n+1, box, 1)) (col (53), p"$$$,$$$,$$$")
("-") (col (66), a)
("Cost/Plant:") (col (68), a)
(cost_per_pl (n, box, 2)) (col (79), p"$$$,$$$,$$$")
("-") (col (93), a)
("Cost/Plant:") (col (94), a)
(cost_per_pl (n+1, box, 2)) (col (105), p"$$$,$$$,$$$")
("-") (col (117), a);
put file (report) skip edit (label (1, n)) (col (5), a)
("-") (col (15), a)

```



```

a);
("XVadd:") (col (16), a)
(impact (n, box, 1)) (col (23), p"z,zzzv.zz")
("-") (col (41), a)
("XVadd:") (col (42), a)
(impact (n+1, box, 1)) (col (49), p"z,zzzv.zz")
("-") (col (66), a)
("XVadd:") (col (68), a)
(impact (n, box, 2)) (col (75), p"z,zzzv.zz")
("-") (col (93), a)
("XVadd:") (col (94), a)
(impact (n+1, box, 2)) (col (100), p"z,zzzv.zz")
("-") (col (117), a);
put file (report) skip edit (label2 (n, box), "- ", "- ", "- ", "- ", "- ");
(col (x (box)), a, col (15), a, col (41), a, col (66), a, col (93), a, col (117),

a);

put file (report) edit (label (2, n)) (col (4), a);
do j = 15 to 117;
  put file (report) edit (" ") (col (j), a);
end;
end;
end;

end;
if "no_report & "tt then do;
  do j = 15 to 115;
    put file (report) edit (" ") (col (j), a);
  end;
  do nstat = 1 to 2;
    do i = 1 to 2;
      if (sum (va (*) ) = 0 & sum (num_pl (*, i, nstat)) = 0) then do;
        vadd (i, nstat) = sum (va (*) ) * sum (num_pl (*, i, nstat)) / sum (num_plants (*));
        pctva (nstat, i) = (sum (costs (*, i, nstat)) / vadd (i, nstat)) * 1e6;
      end;
    else do;
      vadd (i, nstat) = 0;
      pctva (nstat, i) = 0;
    end;
  end;
  if sum (num_pl (*, i, nstat)) = 0 then cpp (i, nstat) =
    (sum (costs (*, i, nstat)) / sum (num_pl (*, i, nstat))) * 1e6;
  else cpp (i, nstat) = 0;
end;

end;
do box = 2 to 1 by -1;
  do i = 1 to 4;
    put file (report) skip edit ("- ", "- ", "- ", "- ") (col (15), a, col (65), a, col (115), a);
  end;

  put file (report) edit ("- ") (col (15), a)
  ("Number of Plants:") (col (27), a)
  (sum (num_pl (*, box, 1))) (col (46), p"zz,zzz")
  ("-") (col (65), a)
  ("Number of Plants:") (col (77), a)
  (sum (num_pl (*, box, 2))) (col (96), p"zz,zzz")
  ("-") (col (115), a);
  put file (report) skip edit ("- ", "- ", "- ", "- ") (col (15), a, col (65), a, col (115), a);
  put file (report) skip edit ("-Plants") (col (5), a)
  ("-") (col (15), a)
  ("Total Annual Cost (MMS):") (col (20), a)
  (sum (costs (*, box, 1))) (col (49), p"$,$$v,$$")
  ("-") (col (65), a)

```

```

col (115). a);

(-Total Annual Cost (WMS):-) (col (70). a)
(sum (costs (+ box. 2))) (col (99). p"$$$$$.SS")
(-) (col (115). a);
if box = 1 then
  put file (report) skip edit ("With", "-|-", "-|-") (col (6). a, col (15). a, col (65). a, col (85). a)
else put file (report) skip edit ("Without", "-|-", "-|-") (col (4). a, col (15). a, col (65). a, col (85). a)

col (115). a);
put file (report) skip edit ("Surface") (col (4). a)
(-|-) (col (15). a)
(-Value Added (WMS):-) (col (26). a)
(vadd (box. 1)) (col (48). p"zz.zzzv.zz")
(-|-) (col (65). a)
(-Value Added (WMS):-) (col (76). a)
(vadd (box. 2)) (col (98). p"zz.zzzv.zz")
(-|-) (col (115). a);

put file (report) skip edit ("Impoundments", "-|-", "-|-") (col (2). a, col (15). a, col (65). a, col (85). a)

col (115). a);
put file (report) skip edit ("-|-") (col (15). a)
(-Cost per Plant:-) (col (29). a)
(cop (box. 1)) (col (45). p"$$$$$.SS")
(-|-) (col (65). a)
(-Cost per Plant:-) (col (79). a)
(cop (box. 2)) (col (95). p"$$$$$.SS")
(-|-) (col (115). a);

put file (report) skip edit ("-|-", "-|-") (col (15). a, col (65). a, col (115). a);
put file (report) edit ("-|-") (col (15). a)
("Impact:-") (col (37). a)
(pctva (1. box). "-X") (col (50). p"z.zzzv.zz", a)
(-|-) (col (65). a)
("Impact:-") (col (87). a)
(pctva (2. box). "-X") (col (100). p"z.zzzv.zz", a)
(-|-) (col (115). a);

do i = 1 to 4;
  put file (report) skip edit ("-|-", "-|-") (col (15). a, col (65). a, col (115). a);
end;
do j = 15 to 115;
  put file (report) edit ("-") (col (j). a (1));
end;

end;

end;
ind_save = industry;
repeat = "1b";
if no_report then do;
  do mod = 1 to num_models;
    do nstat = 1 to 2;
      do i = 1 to 2;
        tnp (mod) = tnp (mod) + tot_pl (mod, i, nstat);
      end;
    end;
  end;
  signal endpage (report);
  put file (report) skip edit ("Total of All Industries") (col (53). a);
  put file (report) skip (2);
  put file (report) skip edit ("Prior State Regulations", "No Prior State Regulations")

```



```

("||") (col (68), a)
("Tot. Cost(MM$):") (col (68), a)
(totcosts (n, box, 2)) (col (84), p"$, $$$v, $$$")
("||") (col (93), a)
("Tot. Cost(MM$):") (col (94), a)
(totcosts (n+1, box, 2)) (col (109), p"$, $$$v, $$$")
("||") (col (117), a);
put file (report) edit ("||") (col (15), a)
("Val. Add(MM$):") (col (16), a)
(value_added (n, box, 1)) (col (31), p"zz, zzzv, zz")
("||") (col (41), a)
("Val. Add(MM$):") (col (42), a)
(value_added (n, box, 1)) (col (56), p"zz, zzzv, zz")
("||") (col (66), a)
("Val. Add(MM$):") (col (68), a)
(value_added (n, box, 2)) (col (83), p"zz, zzzv, zz")
("||") (col (93), a)
("Val. Add(MM$):") (col (94), a)
(value_added (n+1, box, 2)) (col (108), p"zz, zzzv, zz")
("||") (col (117), a);
put file (report) skip edit ("||") (col (15), a)
("Cost/Plant:") (col (16), a)
(cost_per_pl (n, box, 1)) (col (27), p"$, $$$, $$$")
("||") (col (41), a)
("Cost/Plant:") (col (42), a)
(cost_per_pl (n+1, box, 1)) (col (53), p"$, $$$, $$$")
("||") (col (66), a)
("Cost/Plant:") (col (68), a)
(cost_per_pl (n, box, 2)) (col (79), p"$, $$$, $$$")
("||") (col (93), a)
("Cost/Plant:") (col (94), a)
(cost_per_pl (n+1, box, 2)) (col (105), p"$, $$$, $$$")
("||") (col (117), a);
put file (report) edit (label (1, n)) (col (5), a)
("||") (col (15), a)
("XVadd:") (col (16), a)
(impact (n, box, 1), "-X") (col (23), p"z, zzzv, zz", a)
("||") (col (41), a)
("XVadd:") (col (42), a)
(impact (n+1, box, 1), "-X") (col (49), p"z, zzzv, zz", a)
("||") (col (66), a)
("XVadd:") (col (68), a)
(impact (n, box, 2), "-X") (col (75), p"z, zzzv, zz", a)
("||") (col (93), a)
("XVadd:") (col (94), a)
(impact (n+1, box, 2), "-X") (col (101), p"z, zzzv, zz", a)
("||") (col (117), a);
put file (report) skip edit (label2 (n, box), "||", "-||", "-||", "-||")
(col (x (box)), a, col (15), a, col (41), a, col (66), a, col (93), a, col (117), a);
put file (report) edit (label (2, n)) (col (4), a);
do j = 15 to 115;
  put file (report) edit ("--") (col (j), a (1));
end;
/* end j */
/* end n */
/* end box */

```

```

if (sum (tva (*) ) - 0 & sum (tot_pl (*, i, nstat)) - 0) then do;
  vadd (i, nstat) = sum (tva (*)) + sum (tot_pl (*, i, nstat))/sum (tvp (*));
end;
pctva (nstat, i) = (sum (totcosts (*, i, nstat))/vadd (i, nstat))*1e8;
else do;
  pctva (nstat, i) = 0;
  vadd (i, nstat) = 0;
end;
if sum (tot_pl (*, i, nstat)) - 0 then cpp (i, nstat) =
  (sum (totcosts (*, i, nstat))/sum (tot_pl (*, i, nstat)))*1e8;
else cpp (i, nstat) = 0;
end;

do box = 2 to 1 by -1;
  do i = 1 to 4;
    put file (report) skip edit ("i", "i", "i") (col (15), a, col (65), a, col (115), a);

    put file (report) edit ("i") (col (15), a)
      ("Number of Plants:") (col (27), a)
      (sum (tot_pl (*, box, i)) (col (48), p"zz.zzz")
      ("Number of Plants:") (col (77), a)
      (sum (tot_pl (*, box, 2)) (col (99), p"zz.zzz")
      ("i") (col (115), a);

    put file (report) skip edit ("i", "i", "i") (col (15), a, col (65), a, col (115), a);
    put file (report) skip edit ("Plants") (col (5), a)
      ("Total Annual Cost (MM$):") (col (20), a)
      (sum (totcosts (*, box, 1)) (col (49), p"$, $$$v.$$")
      ("Total Annual Cost (MM$):") (col (70), a)
      (sum (totcosts (*, box, 2)) (col (99), p"$, $$$v.$$")
      ("i") (col (115), a);

    if box = 1 then
      put file (report) skip edit ("With", "i", "i", "i") (col (8), a, col (15), a, col (65), a, col (115), a);

    else put file (report) skip edit ("Without", "i", "i", "i") (col (4), a, col (15), a, col (65), a, col (115), a);

    put file (report) skip edit ("Surface") (col (4), a)
      ("i") (col (15), a)
      ("Value Added (MM$):") (col (26), a)
      (vadd (box, 1)) (col (48), p"zz.zzzv.zz")
      ("i") (col (65), a)
      ("Value Added (MM$):") (col (76), a)
      (vadd (box, 2)) (col (98), p"zz.zzzv.zz")
      ("i") (col (115), a);

    put file (report) skip edit ("Impoundments", "i", "i", "i") (col (2), a, col (15), a, col (65), a, col (115), a);

    col (115), a);
    put file (report) skip edit ("i") (col (15), a)
      ("Cost per Plant:") (col (29), a)
      (cpp (box, 1)) (col (45), p"$, $$$, $$$")
      ("i") (col (65), a)
      ("Cost per Plant:") (col (79), a)
      (cpp (box, 2)) (col (95), p"$, $$$, $$$")
      ("i") (col (115), a);

    put file (report) skip edit ("i", "i", "i") (col (15), a, col (65), a, col (115), a);
  
```



```

put file (report) edit (";-") (col (15), a)
("impact;-") (col (37), a)
(pctva (1, box), "-x-") (col (50), p"x,zzzv.zz", a)
(";-") (col (65), a)
("impact;-") (col (87), a)
(pctva (2, box), "-x-") (col (100), p"x,zzzv.zz", a)
(";-") (col (115), a);

do i = 1 to 4;
  put file (report) skip edit ("|-", "|-", "|-") (col (15), a, col (65), a, col (115), a);
end;
do j = 15 to 115;
  put file (report) edit ("--") (col (j), a (1));
end;
end;
end;
put file (report) skip;
signal endpage (report);
end;
close file (report);
close file (datafile);
end box;

```

```

buc_report: procedure;
Xinclude rcra;

dcl output (5, 5, num_segments+hi_incr, ncol) float dec (6) based (optr);
indsum (num_segments+num_hi_seg, 5, 5, ncol) float dec based (is_ptr);
(frec, smt, mod, i, ii, j, nsec, counter, alen, argno init (1), num_models static init (4), hi_incr init (0)) fixed bi
n,
ncol static fixed bin init (9),
ec fixed bin (35),
col_name (ncol+1) char (11) var init ("Tech Reqt", "Disposal", "Finl Reqt", "Rec/Rep", "Mon/Test", "Admin", "Training
",
"Contingency", "Non-ded F/R", "Row Total");
(str2, industry, ind_save, old_ind, segment) char (60) var;
yr_type (5) char (10) var init ("Initial", "Annual", "Capital", "Closure", "Annualized");
arg char (alen) unaligned based (aptr);

dcl (month, dom, year, hour, minute, second, dow) fixed bin,
(bits, microsecond) fixed bin (71),
code fixed bin (35),
sun fixed bin (35),
filler char (3) var init (1),
zone char (3) var init ("--"),
shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT"),
mo (12) char (10) var init ("January", "February", "March", "April", "May", "June",
"July", "August", "September", "October", "November", "December"),
days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");

dcl line_num fixed bin,
page_str char (3) var,
pageno pic"zz9" init (000);
dcl did_rep bit (1) aligned init ("0-b"),
(hisw, industry_sw) bit (1) aligned init ("0-b"),
(more_smt, repeat) bit (1) aligned init ("1-b");

dcl (aptr, optr, is_ptr) pointer;

dcl (ltrim, char, sum, lineno, index) builtin;

dcl (sysprint output, report print output) file stream,
cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35)),
annualize_npv_ext entry (float dec (6), float dec (6), float dec (6), float dec (6)),
decode_clock_value_date_time entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71)),
fixed bin, char (3), fixed bin (35));
dcl header_ext entry (char (64) var, file);
dcl endpage condition;

Xpage;
on endpage (report) begin;
line_num = lineno (report);
do while (line_num < 59);
put file (report) skip;

```

```

line_num = lineno (report);
end;
page_str = trim (char (pageno));
put file (report) edit ("PAGE", page_str) (col (125), a (4), x (1), a);
put file (report) page;
put file (report) skip;
put file (report) edit (mo (month), dom, " ", year, hour, " ",
filler, trim (char (minute)), shift (sun), days (dow))
(col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
x (1), f (2), a (1), a, x (1), a, x (1), a (3));
call header_ ("MASTER REPORT", report);
put file (report) skip (2);
pageno = pageno + 001;
end;
/* end on endpage */

xpage:
arglp:
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to proceed;
else if arg = "-himpact" | arg = "-hl" then hlsw = "1'b";
else if arg = "-industry" | arg = "-ind" then industry_sw = "1'b";
else do;
put skip list ("Args are: -himpact -industry");
put skip (2);
return;
end;
argno = argno + 1;
go to arglp;

```

```

proceed:
open file (report) title ("file_master.report") print linesize (135);
if hlsw then hl_incr = num_hl_seg;
allocate output, indsum;

bits = clock ();
call decode_clock_value ($date_time
(bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code));
if hour = 12 then do;
if minute = 0 then sun = 3;
else sun = 2;
end;
else if hour = 0 then do;
hour = 12;
if minute = 0 then sun = 4;
end;
else if hour > 12 then do;
hour = hour - 12;
sun = 2;
end;
if minute < 10 then filler = "0";

output (*, *, *, *) = 0.0;
indsum (*, *, *, *) = 0.0;
do sat = 1 to num_segments+hl_incr;
do nsec = 1 to 5;
do fsec = 1 to 4;
do mod = 1 to num_models;
do i = 1 to ncol-1;
if i = 1 then ii = 1;
else ii = i+1;
if sat <= num_segments then

```

```

output (freq, nsec, smt, i) =
output (freq, nsec, smt, i) + (carrays$ (smt, 2, mod, nsec, freq, i) +
state_regs (nsec, i) + arrays$ (smt, 1, mod, nsec, freq, i)) / 1e6;
else
output (freq, nsec, smt, i) =
state_regs (nsec, i) + impacts$ (smt, 2, mod, nsec, freq, i) +
/* end i */
/* end mod */
/* end freq */
/* end nsec */
end;

output (2, 4, smt, 2) = output (2, 4, smt, 1);
output (2, 4, smt, 1) = 0e0;

/* end smt */

do nsec = 1 to 5;
do smt = 1 to num_segments + hi_incr;
do i = 1 to ncol;
output (5, nsec, smt, i) = annualize_npv_ (
(output (1, nsec, smt, i)),
(output (2, nsec, smt, i)),
(output (3, nsec, smt, i)),
(output (4, nsec, smt, i)));
end;
end;

/* end i loop */
/* end nsec, smt */

signal endpage (report);
smt = 1;
str2 = ind_struc$ (smt).ind_name;
industry = substr (str2, 1, index (str2, ".") - 1);
ind_save = industry;
counter = 1;
do while (more_smt);
do freq = 1 to 5;
do nsec = 1 to 5;
do i = 1 to ncol;
indsum (counter, freq, nsec, i) =
/* end i */
/* end nsec */
/* end freq */
end;
end;

smt = smt + 1;
old_ind = str2;
if smt <= num_segments + hi_incr then do;
str2 = ind_struc$ (smt).ind_name;
industry = substr (str2, 1, index (str2, ".") - 1);
else more_smt = "0";
repeat = industry_sw;
end;
if (industry_sw) then put file (report) edit (ind_save) (col (65 - length (ind_save)/2), a);
else do;
segment = substr (old_ind, index (old_ind, ".") + 1);
put file (report) skip edit (ind_save) (col (65 - length (ind_save)/2), a);
(segment) (col (65 - length (segment)/2), a);
end;
put file (report) skip (3);
put file (report) edit (col_name (1)) (col (12), a)
(col_name (2)) (col (23), a)
(col_name (3)) (col (36), a)

```

```

(col_name (4)) (col (47), a)
(col_name (5)) (col (61), a)
(col_name (6)) (col (72), a)
(col_name (7)) (col (82), a)
(col_name (8)) (col (95), a)
(col_name (9)) (col (109), a)
(col_name (10)) (col (122), a);
put file (report) skip (2);
do freq = 1 to 5;
  put file (report) list (yr_type (freq));
  do j = 1 to 5;
    put file (report) skip edit
      ('300' || trim (char (j))) || '-'; (indsum (counter, freq, j, 1)) do i = 1 to ncol);
    sum (indsum (counter, freq, j, *));
    (a (5), col (9), (ncol+1) (x (3), p"x.zzzv.zzz"));
    /* end of j loop */
  end;
  put file (report) skip (2);
  /* end of freq loop */
end;
do freq = 1 to 4;
  put file (report) skip edit
    (yr_type (freq), '-'); (sum (indsum (counter, freq, *, 1)) do i = 1 to ncol);
  sum (indsum (counter, freq, *, *)); (r (1));
  format
    (a, a (1), col (9), (ncol+1) (x (3), p"x.zzzv.zzz"));
  /* end of freq loop */
end;

put file (report) skip edit
  (yr_type (5), '-'); (sum (indsum (counter, 5, *, 1)) do i = 1 to ncol);
sum (indsum (counter, 5, *, *));
(r (1));
signal endpage (report);
ind_save = industry;
counter = counter + 1;
repeat = '1'b;

end;

/* print summary page for all industries */
put file (report) edit ('Total of All Industries') (col (50), a);
put file (report) skip (2);
put file (report) edit (col_name (1)) (col (12), a)
  (col_name (2)) (col (23), a)
  (col_name (3)) (col (36), a)
  (col_name (4)) (col (47), a)
  (col_name (5)) (col (61), a)
  (col_name (6)) (col (72), a)
  (col_name (7)) (col (82), a)
  (col_name (8)) (col (95), a)
  (col_name (9)) (col (109), a)
  (col_name (10)) (col (122), a);
put file (report) skip (2);
do freq = 1 to 5;
  put file (report) list (yr_type (freq));
  do j = 1 to 5;
    put file (report) skip edit
      ('300' || trim (char (j))) || '-'; (sum (indsum (*, freq, j, 1)) do i = 1 to ncol);
    sum (indsum (*, freq, j, *));
    (a (5), col (9), (ncol+1) (x (3), p"x.zzzv.zzz"));
    /* end of j loop */
  end;
  put file (report) skip (2);
  /* end of freq loop */
end;

```



```

do freq = 1 to 4;
  put file (report) skip edit (yr_type (freq), ":") (a, a (1));
  put file (report) skip edit
  (" ", " ", sum (indsun (*, freq, *, 1)) do i = 1 to ncol),
  sum (indsun (*, freq, *, *))
  (r (fi));
end;
/* end of freq loop */

put file (report) skip edit (yr_type (5), ":") (a, a (1));
put file (report) skip edit (" ", " ", sum (indsun (*, 5, *, i)) do i = 1 to ncol),
sum (indsun (*, 5, *, *))
(r (fi));

signal endpage (report);
close file (report);
free output, indsum;

end buc_report;

```

I-35

end closure_1

```

/* .....
/*
/* Written in 1980 at:  Arthur D. Little, Inc.
/*                      Cambridge, Mass. 02140
/*
/* .....
/*
/* Written 6-80 by K. A. Speelman
/* .....
/*

cpf: procedure;

#include rcra;

dcl list (counter) based (list_ptr) fixed bin;
dcl list_ptr pointer;

dcl 1 rep_struc (num_segments + hi_incr) aligned based (rep_ptr).
    2 name char (60).
    2 annual_cost float dec.
    2 annual_waste float dec.
    2 c1cost float dec.
    2 fac float dec;

dcl  outputs (4, 5, 124, 8) ext static fixed dec (9, 3).
    (cap, c1cost, tve, aoe, od, tac, cost_per_fac, cost_per_waste, va, fac, tau) float dec.
    (ttac, ttaw, ttac) float dec;
    (gttac, gttaw, gttac) float dec;
dcl  (frec, smt, mod, i, j, nsec, alen, argno init (1), num_models static init (4)) fixed bin,
    hi_incr fixed bin init (0).
    counter fixed bin init (0).
    ec fixed bin (35).
    (month, dom, year, hour, minute, second, dow) fixed bin,
    (bits, microsecond) fixed bin (71).
    code fixed bin (35).
    sun fixed bin init (1);
    str1 char (64) var init ("Cost per Facility").
    arg char (alen) unaligned based (aptr),
    option char (15) var,
    opstr char (65) var init ("tot_ann_cost tot_ann_waste nfac cost_per_fac cost_per_waste").
    (str2, old_ind, industry, ind_save) char (60) var,
    filler char (3) var init (" ").
    zone char (3) init (" ").
    shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT").
    mo (12) char (10) var init ("January", "February", "March", "April", "May", "June", "July",
    "August", "September", "October", "November", "December").
    days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");
dcl  (more_smt, repeat) bit (1) aligned init ("1'b");
dcl  (h1sw, industry_sw, order, up) bit (1) aligned init ("0'b");
dcl  (aptr, optr, rep_ptr) pointer;

dcl  (substr, index, sum, length) builtin,
    endpage condition.

```

```

(sysprint output, report output) file stream,
cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35)),
annualize_npv_ext entry (float dec, float dec, float dec, float dec) returns (float dec),
header_ext entry (char (64) var, file);
dc1 decode_clock_value_sdate_time entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
fixed bin, char (3), fixed bin (35));
xpage:
on endpage (report) begin;
  put page file (report) edit (mo (month), dom, "...", year, hour, "...",
  filler, trim (char (minute))), shift (sun), days (dow))
  (col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
  x (1), f (2), a (1), a, x (1), a, x (1), a (3));
  put file (report) skip;
  call header_ (str1, report);
  put file (report) skip (2);
  if -industry_sw then put file (report) edit ("Industry:") (col (5), a);
  put file (report) edit ("Total Annual") (col (39), a)
  ("Number of") (col (58), a)
  ("Cost per") (col (78), a)
  ("Cost per") (col (98), a)
  ("Cost per") (col (117), a);
  put file (report) skip;
  if industry_sw then put file (report) edit ("Industry") (col (5), a);
  else put file (report) edit ("Segment") (col (5), a);
  put file (report) edit ("Cost") (col (43), a)
  ("Waste") (col (61), a)
  ("Facilities") (col (78), a)
  ("Facility") (col (98), a)
  ("MT Waste") (col (117), a);
  put file (report) skip;
  if industry_sw then put file (report) edit ("Industry") (col (5), a);
  else put file (report) edit ("Segment") (col (5), a);
  put file (report) edit ("Cost") (col (43), a)
  ("Waste") (col (61), a)
  ("Facilities") (col (78), a)
  ("Facility") (col (98), a)
  ("MT Waste") (col (117), a);
  put file (report) skip (2);
end;

/* begin block */

arglp:
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to proceed;
else if arg = "-impact" then h1sw = "1";
else if arg = "-industry" then h1sw = "1";
else if arg = "-sort" then do;
  order = "1";
  argno = argno + 1;
  call cu_sarg_ptr (argno, aptr, alen, ec);
  if ec = 0 then option = "cost_per_waste";
  else if index (arg, "-") = 0 then do;
    argno = argno - 1;
    option = "cost_per_waste";
  end;
else if index (opstr, arg) = 0 then do;
  put skip edit ("Sort options are:", opstr) (a, x (2), a);
  put skip (2);
end;

```



```

        return;
    end;
    else option = arg;
    end;
    else if arg = "--asc" | arg = "--desc" then do;
        if "order" then do;
            put skip list ("If sort desired, list as argument.");
            put skip list ("Args -desc and -asc invalid without sort arg.");
            put skip (2);
            return;
        end;
        else if arg = "--asc" then up = "1b";
    end;
    else do;
        put skip list ("Args are: -himpact -industry -sort <option> -desc -asc");
        put skip (2);
        return;
    end;
    argno = argno + 1;
    go to argip;
/* end error do */

proceed:
    if hlsw then hl_incr = num_hl_seg;
    allocate rep_struc;
    open file (report) title ("vfile_cpf.report") print linesize (135);
    bits = clock ();
    call decode_clock_value_date_time
        (bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code);
    if hour = 12 then do;
        if minute = 0 then sun = 3;
        else sun = 2;
    end;
    else if hour = 0 then do;
        hour = 12;
        if minute = 0 then sun = 4;
    end;
    else if hour > 12 then do;
        hour = hour - 12;
        sun = 2;
    end;
    if minute < 10 then filler = "0";

    outputs (" ", " ", " ") = 0.0;
    do smt = 1 to num_segments+hl_incr;
        rep_struc (smt).annual_cost = 0e0;
        rep_struc (smt).annual_waste = 0e0;
        rep_struc (smt).cicost = 0e0;
        rep_struc (smt).fac = 0e0;
        do nsec = 1 to 5;
            do freq = 1 to 4;
                do i = 1 to 8;
                    do mod = 1 to num_models;
                        if smt <= num_segments then
                            outputs (freq, nsec, smt, i) =
                                outputs (freq, nsec, smt, i) +
                                    (carrys$ (smt, 2, mod, nsec, freq, i) +
                                        state_regs (nsec, i) * carrys$ (smt, 1, mod, nsec, freq, i)) / 1e8;
                            else
                                outputs (freq, nsec, smt, i) =
                                    outputs (freq, nsec, smt, i) +
                                        (impact$ (smt, 2, mod, nsec, freq, i) +
                                            state_regs (nsec, i) * impact$ (smt, 1, mod, nsec, freq, i)) / 1e8;

```

```

end; end; end; end; end; /* end mod */

signal endpage (report);
ttac, ttaw, tfac = 0e0;
gttac, gtaw, gtfac = 0e0;
smt = 1;
str2 = ind_struct$ (smt).ind_name;
industry = substr (str2, 1, index (str2, "-")-1);
ind_save = industry;
do while (more_smt);
  counter = counter + 1;
  if industry_sw then rep_struct (counter).name = industry;
  else rep_struct (counter).name = str2;
  do while (industry = ind_save & more_smt & repeat);
    va = sum (ind_struct$ (smt).value, add (*));
    cap = sum (outputs$ (3, *, smt, *));
    fac = sum (stat_block$ (smt).size (*).gen_ons (*)) * sum (stat_block$ (smt).size (*).waste_ons (*));
    fye = sum (outputs$ (1, *, smt, *));
    clocost = sum (outputs$ (4, *, smt, *));
    aoc = sum (outputs$ (2, *, smt, *));
    aoe = aoc - outputs$ (2, 4, smt, 1);
    od = outputs$ (2, 4, smt, 1);
    tac = annualize_npv_ (fye, aoe+od, cap, clocost);
    taw = sum (stat_block$ (smt).size (*).waste_offs (*)) + sum (stat_block$ (smt).size (*).waste_ons (*));
    rep_struct (counter).annual_cost = rep_struct (counter).annual_cost + tac;
    rep_struct (counter).annual_waste = rep_struct (counter).annual_waste + taw;
    rep_struct (counter).clocost = rep_struct (counter).clocost + clocost;
    smt = smt + 1;
    /* not recurring tech */
    old_ind = str2;
    if smt <= num_segments + hi_incr then do;
      str2 = ind_struct$ (smt).ind_name;
      industry = substr (str2, 1, index (str2, "-")-1);
    end;
    else more_smt = "0";
    repeat = industry_sw;
  end;
  ind_save = industry;
  repeat = "1";
end;
allocate list;
if order then call sort (option, up);
else do i = 1 to counter;
  list (i) = i;
end;

do i = 1 to counter;
  if rep_struct (list (i)).fac = 0 then cost_per_fac = rep_struct (list (i)).annual_cost/rep_struct (list (i)).
  else cost_per_fac = 0;
  if rep_struct (list (i)).annual_waste = 0 then cost_per_waste = rep_struct (list (i)).annual_cost/
  else cost_per_waste = 0;
  if (industry_sw) then put file (report) skip edit
  (rep_struct (list (i)).name) (col (1), a (28));
  else put file (report) skip edit (substr
  (rep_struct (list (i)).name, 1, index (rep_struct (list (i)).name, "-")))
  (a)

```

```

(substr (rep_struct (list (1)).name, index (rep_struct (list (1)).name, ":") +2))
(skip, a (35));
put file (report) edit
(rep_struct (list (1)).annual_cost + 1e6) (col (37), p"$$. $$$. $$$.$$$")
(rep_struct (list (1)).annual_waste) (col (59), p"zz.zzz.zzz")
(rep_struct (list (1)).fac) (col (77), p"zz.zzz.zzz")
(cost_per_fac) (col (95), p"$$. $$$. $$$.$$$")
(cost_per_waste) (col (111), p"$$. $$$. $$$.$$$");
end;

```

/* print summary page for all industries */

```

put file (report) skip;
if sum (rep_struct (+).fac) /= 0 then cost_per_fac =
  (sum (rep_struct (+).annual_cost))/
  (sum (rep_struct (+).fac)) * 1e6;
else cost_per_fac = 0;
if sum (rep_struct (+).annual_waste) /= 0 then cost_per_waste =
  sum (rep_struct (+).annual_cost)/
  sum (rep_struct (+).annual_waste) * 1e6;
else cost_per_waste = 0;
put file (report) skip edit
("total") (col (1), a (28))
(sum (rep_struct (+).annual_cost) + 1e6) (col (36), p"$$. $$$. $$$.$$$")
(sum (rep_struct (+).annual_waste)) (col (60), p"zz.zzz.zzz")
(sum (rep_struct (+).fac)) (col (78), p"zz.zzz.zzz")
(cost_per_fac) (col (95), p"$$. $$$. $$$.$$$")
(cost_per_waste) (col (112), p"$$. $$$. $$$.$$$");
put file (report) skip;
close file (report);
free rep_struct;
free list;

```

```

sort;
procedure (option, up);
dcl option char (15) var;
dcl up bit (1) aligned;
dcl item (counter) float dec;
dcl bound float dec;
dcl (i, j, k) fixed bin;
if option = "tot_ann_cost" then do i = 1 to counter;
  item (i) = rep_struct (i).annual_cost;
end;
else if option = "tot_ann_waste" then do i = 1 to counter;
  item (i) = rep_struct (i).annual_waste;
end;
else if option = "nfac" then do i = 1 to counter;
  item (i) = rep_struct (i).fac;
end;
else if option = "cost_per_fac" then do i = 1 to counter;
  if rep_struct (i).fac /= 0 then
    item (i) = rep_struct (i).annual_cost/rep_struct (i).fac;
  else item (i) = 0e0;
end;
else if option = "cost_per_waste" then do i = 1 to counter;
  if rep_struct (i).annual_waste /= 0 then
    item (i) = rep_struct (i).annual_cost/rep_struct (i).annual_waste;
end;

```

```

else item (i) = 0e0;
end;
bound = item (i);
k = 1;
do i = 1 to counter;
do j = 1 to counter;
if (up & item (j) < bound) | (~up & item (j) > bound) then do;
bound = item (j);
k = j;
end;
end;
list (i) = k;
if up then do;
item (k) = 1000000000;
bound = 1000000000;
end;
else do;
item (k) = -1;
bound = -1;
end;
end;
return;
end sort;
end cpi;

```

```

/* .....
/* Written in 1980 at Arthur D. Little, Inc.
/* Cambridge, Mass.
/* .....
/* .....
/* first coded by K. A. Speelman 6-80
/* modified 9-80 by T. J. Martin for waste groups
/* .....

data_report: procedure;
dcl (sysprint output, datafile input, datafile2 input, prtfile print output) file stream,
endfile condition, endpage condition,
(length, substr, lineno, trim, char, index, clock) builtin;
dcl decode_clock_value $date_time entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
fixed bin, char (3), fixed bin (35));

dcl (i, p, j, x, line_num, month, dow, year, hour, minute, second, dow)
fixed bin,
(bits, microsecond) fixed bin (71),
code fixed bin (35);
dcl pageno pic'zz9' init (000),
page_str char (3) var;
dcl true bit (1) aligned init ('1'b);
dcl (name, ind_name, segment, industry) char (80) var,
(nplants, nws, nwgr) fixed bin,
(vol, emp, np, vadd, pr, pct1) float dec,
mname char (3) var;

dcl reg (2) char (13) var init ('Regulated', 'Not Regulated'),
cutoff (3) char (4) var init ('100', '500', '1000'),
sic char (5) var,
filler char (3) var init ('-'),
zone char (3) init ('-'),
shift (4) char (8) var init ('AM', 'PM', 'NOON', 'MIDNIGHT');
dcl mo (12) char (10) var init ('January', 'February', 'March', 'April', 'May',
'June', 'July', 'August', 'September', 'October', 'November', 'December');
dcl days (7) char (3) var init ('Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat',
'Sun');
dcl sun fixed bin init (1),
volume float dec,
wname char (60) var,
cprac (3) fixed bin,
tech (14) fixed bin,
gpercent float dec,
cpct (3) float dec,
nprac (2) fixed bin,
npct (2) float dec,
micode char (1) var,
pct_ons float dec;

```



```

dcl old_prac (0:31) char (4) var
  init ("", "Lofl", "SLfl", "DWl", "Burn",
        "Stor", "Recy",
        "Recv", "POIW", "LFID", "LFRm",
        (3) (1)--, "Dump", (4) (1)--,
        "Inc", "Incl", "Incl", "Inc2",
        (7) (1)--, "Unlg");

dcl new_prac (0:130) char (4) var
  init ((101) (1)--, "Ldfl", "SLfl", "DWl",
        (3) (1)--, "Recy", (2) (1)--, "SLfD",
        "LFRm", (3) (1)--, "Ldfl", (4) (1)--,
        "Inc", "Incl", "Incl", "Inc2",
        "Inc3", "Inc3", (4) (1)--, "Legn");

dcl al_present fixed bin,
  (no_imp, disp_imp, ts_imp, both_imp, num_disp_pp, disp_sa, num_ts_pp,
   ts_sa) float dec;
dcl size (4) char (8) var init ("LARGE:", "MEDIUM:", "SMALL:", "SWG:");
xpage;
  on endfile (datafile) begin;
    close file (datafile);
    close file (datafile2);
    go to done;
  end;

  on endpage (prtfille) begin;
    line_num = lineno (prtfille);
    do while (line_num < 59);
      put file (prtfille) skip;
      line_num = lineno (prtfille);
    end;
    page_str = ltrim (char (pageno));
    put file (prtfille) edit ("PAGE", page_str) (col (125), a (4), x (1), a);
    put file (prtfille) page;
    put file (prtfille) skip (1);
    pageno = pageno + 001;
  end;

  open file (datafile) title ("vfile_adl.data");
  open file (datafile2) title ("vfile_sia.data");
  open file (prtfille) title ("vfile_data.report") linesize (132);

  bits = clock ();
  call decode_clock_value ($date_time
    {bits, month, day, year, hour, minute, second, microsecond, dow, zone, code});
  if hour = 12 then do;
    if minute = 0 then sun = 3;
    else sun = 2;
  end;
  if hour = 0 then do;
    hour = 12;
    if minute = 0 then sun = 4;
  end;
  else if hour > 12 then do;
    hour = hour - 12;
    sun = 2;
  end;
  if minute < 10 then filler = "0";

```

```

signal endpage (prtfile);
do while (true);
  get file (datafile) list (ind_name, vadd, nplants, nws, nwgr);
  get file (datafile2) list (name, sic, si_present);
  industry = substr (ind_name, 1, index (ind_name, "-") - 1);
  segment = substr (ind_name, index (ind_name, "-") + 2);
  p = length (sic);
  do while (p < 4);
    sic = sic || "x";
    p = length (sic);
  end;

  /* end do while */

  put file (prtfile) edit (mo (month), dom, " ", year, hour, ":",
    filler, ltrim (char (minute)), shift (sun), days (dow))
    (col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
    x (1), f (2), a (1), a, x (1), a, x (1), a (3));
  put file (prtfile) skip (2);
  put file (prtfile) edit
    (industry) (col (65 - length (industry) / 2), a)
    (segment) (skip, col (65 - length (segment) / 2), a);
  put file (prtfile) skip edit (sic) (col (65 - length (sic) / 2), a);
  put file (prtfile) skip (2);
  put file (prtfile) skip edit ("VALUE ADDED (NM$): ", vadd)
    (col (15), a, f (5));

  put file (prtfile) skip edit ("NUMBER OF MODELS: ", nplants*2)
    (col (15), a, f (2));
  put file (prtfile) skip edit ("NUMBER OF WASTE STREAMS: ", nws)
    (col (15), a, f (3));
  put file (prtfile) skip edit ("NUMBER OF WASTE GROUPS: ", nwgr)
    (col (15), a, x (1), p"z9");
  put file (prtfile) skip;

  /* print plant information */
  put file (prtfile) skip edit ("PLANT INFORMATION") (a);
  put file (prtfile) skip edit ("-----") (a);
  put file (prtfile) skip edit ("Production") (col (41), a);
  put file (prtfile) skip edit ("Number", "per plant",
    "Employment", "Waste per", "Percent")
    (col (30), a, col (41), a, col (55), a, col (68), a,
    col (84), a);
  put file (prtfile) skip edit ("Size", "State Group", "of Plants", "(000 MT)", "per plant", "Plant (MT)",
    "Level I") (col (1), a, col (10), a, col (28), a, col (42), a, col (55), a, col (68), a,
    col (84), a);
  put file (prtfile) skip edit ("-----") (col (1), a, col (10), a, col (28), a, col (41), a, col (55), a, col (68), a, col (84), a);
  do j = 1 to 2;
    do i = 1 to nplants;
      get file (datafile) list (mname, np, pr, emp, vol, pctil);
      if mname = "L" then x = 1;
      else if mname = "M" then x = 2;
      else if mname = "S" then x = 3;
      else x = 4;
      put file (prtfile) skip edit (size (x), reg (j), np, pr, emp, vol, pctil * 100)
        (col (1), a, col (10), a, col (31), f (4, 0), col (41), f (10, 2),
        col (60), f (5, 0), col (68), f (10, 2), col (85), p"z9v.99");
      /* end plant loop */
    end;
  end;

  /* end reg/nonreg loop */

```

```

/* print plant SWG cutoff information */
put file (prtfil) skip (7);
put file (prtfil) skip edit ("CUTOFF INFORMATION") (col (1), a);
put file (prtfil) skip edit ("-----") (a);
put file (prtfil) skip;
put file (prtfil) skip edit ("Category", "Cumulative", "Cumulative", "Cumulative")
(col (1), a, col (16), a, col (31), a, col (45), a);
put file (prtfil) skip edit ("(kg/month)", "Waste (MT)",
(col (1), a, col (16), a, col (31), a, col (47), a);
put file (prtfil) skip edit ("-----", "-----")
(col (1), a, col (16), a, col (31), a, col (45), a);
put file (prtfil) skip;
do i = 1 to 3;
  get file (datafil) list (vol, emp, np);
  put file (prtfil) skip edit (cutoff (1), ":", vol, emp, np)
  (col (3), a, col (7), a, col (19), f (6, 0), col (33), f (5, 0), col (46), f (6, 0));
  put file (prtfil) skip;
end;
/* end cutoff loop */

/* print information on all waste streams */
put file (prtfil) skip (7);
put file (prtfil) skip edit ("WASTE STREAM INFORMATION") (col (1), a);
put file (prtfil) skip (2) edit ("-----") (col (1), a);
put file (prtfil) skip edit ("CURRENT WASTE", "NEW WASTE")
(col (50), a, col (86), a);
put file (prtfil) skip edit ("MANAGEMENT PRACTICES", "MANAGEMENT PRACTICES",
"MONITORING") (col (47), a, col (81), a, col (107), a);
put file (prtfil) skip edit ("QUANTITY", "AND TESTING", "PERCENT") (col (27), a, col (106), a, col (119), a);

a);
put file (prtfil) skip edit ("NAME", "(MT)", "Practice (%)", "Practice (%)",
(col (11), a, col (29), a, col (36), a, col (50), a, col (64), a, col (78), a, col (92), a,
col (110), a, col (119), a);
put file (prtfil) skip edit ("-----", "-----", "-----", "-----",
(col (11), a, col (27), a, col (36), a, col (78), a, col (106), a, col (119), a);
put file (prtfil) skip;
do i = 1 to nws;
  get file (datafil) list (wname, volume,
(cprac (j), cpct (j) do j = 1 to 3),
(nprac (j), npct (j) do j = 1 to 3),
mtcode, pct_ons);
  put file (prtfil) skip edit (wname, volume,
(old_prac (cprac (j)), (cpct (j) * 100) do j = 1 to 3),
(new_prac (nprac (j)), (npct (j) * 100) do j = 1 to 3),
mtcode, pct_ons * 100) (col (1), a (22), col (24), f (10, 2), col (38), a (4), col (44), f (3), c
ol (52), a (4),
(4),
col (58), f (3), col (66), a (4), col (72), f (3), col (80), a (4), col (86), f (3), col (94), a
(4),
col (99), f (3), col (111), a, col (120), f (3));
  end;
/* end waste loop */

/* print information on waste groups and allocations */
put file (prtfil) skip (7);
put skip file (prtfil) edit ("WASTE GROUP INFORMATION") (col (1), a);
put skip file (prtfil) edit ("-----") (col (1), a);
put skip file (prtfil) edit ("Group", "Percent", "List of")
(col (1), a, col (10), a, col (20), a);
put skip file (prtfil) edit ("No.", "Total", "Disposal Codes ->")

```

```

(col (1), a, col (10), a, col (20), a);
put skip file (prtfil) edit ("-----", "-----")
(col (1), a, col (10), a, col (20), a);
put skip file (prtfil);
do i = 1 to nwr;
  get file (datafil) list (percent, (tech (j) do j = 1 to nws));
  put file (prtfil) edit (i, percent, (tech (j) do j = 1 to nws))
    (col (2), p"29", col (11), f (4, 2), col (20), (nws) (p"29", x (2)));
end;

/* print surface impoundment information */
put file (prtfil) skip (7);
put file (prtfil) skip edit ("SURFACE IMPOUNDMENT INFORMATION") (a);
put file (prtfil) skip edit ("-----") (a);
put file (prtfil) skip;

/* if sl_present=1 then do; */
put file (prtfil) skip edit ("Size", "State Group", "1", "2", "3", "4", "5", "6", "7", "8")
  (col (1), a, col (10), a, col (30), a, col (40), a, col (50), a, col (60), a, col (70), a,
  col (80), a, col (90), a, col (100), a);
put file (prtfil) skip edit ("-----", "-----", "-----", "-----", "-----", "-----", "-----", "-----")
  (col (1), a, col (10), a, col (30), a, col (40), a, col (50), a, col (60), a, col (70), a, col (80), a,
  col (90), a, col (100), a);
put file (prtfil) skip;
do j = 1 to 4;
  do i = 1 to 2;
    get file (datafil2) list (no_imp, disp_imp, ts_imp, both_imp,
      num_disp_pp, disp_aa, num_ts_pp, ts_aa);
    if (no_imp + disp_imp + ts_imp + both_imp) > 0 then do;
      put file (prtfil) skip edit (size (j), reg (1), no_imp, disp_imp,
        ts_imp, both_imp, num_disp_pp, disp_aa, num_ts_pp, ts_aa)
        (col (1), a (6), col (10), a, col (27), f (5), col (37), f (5),
        col (47), f (5), col (57), f (5), col (68), f (5, 2), col (77), f (5, 2),
        col (88), f (5, 2), col (97), f (6, 2));
      /* end sum do */
    end;
    /* end i loop */
  end;
  /* end j loop */
end;

put file (prtfil) skip (2);
put file (prtfil) skip edit ("(1)", "Number of plants with no impoundments")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(2)", "Number of plants with disposal only")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(3)", "Number of plants with treatment/storage only")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(4)", "Number of plants with both")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(5)", "Number of disposal impoundments per plant")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(6)", "Surface area of disposal impoundments per site")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(7)", "Number of treatment/storage impoundments per plant")
  (col (30), a, col (45), a);
put file (prtfil) skip edit ("(8)", "Surface area of treatment/storage impoundments per site")
  (col (30), a, col (45), a);

else do;
  put file (prtfil) skip;
  put file (prtfil) skip edit ("NO SURFACE IMPOUNDMENTS") (a);
end;

```

```
end; */  
done:   end;   signal endpage (prtfila);  
        end data_report;  
/* end industry loop */
```


exhibit: procedure;

include rcra;

dc1 1 rep_struc (num_segments + hl_incr) aligned based (rep_ptr),

2 name char (60) var,
2 tva float dec (6),
2 tcap float dec (6),
2 tcloct float dec (6),
2 tye float dec (6),
2 taoc float dec (6),
2 tod float dec (6),
2 tfe float dec (6),
2 ttac float dec (6);

dc1 list (counter) based (list_ptr) fixed bin;

dc1 clos_period float dec ext static;

dc1 ann_clos float dec (6);

dc1 (list_ptr, rep_ptr) pointer;

(cap, clocst, fye, tfe, aoe, od, tac, pctva, va, num_gen, alloc, clos_pat) float dec (6),

(tcap, taoc, tod, ttac, tva) float dec (6),

(gtcap, gtfe, gtaoc, gtod, gttac, gttva) float dec (6);

dc1 rt (20) float dec init ((5)1.85e-1, (15)1.78e-1);

dc1 (frec, smt, mod, l, j, nsec, alen, argno init (1), num_models static init (4)) fixed bin,

hl_incr fixed bin init (0),

counter fixed bin init (0),

ec fixed bin (35),

(month, dom, year, hour, minute, second, dow) fixed bin,

(bits, microsecond) fixed bin (71),

code fixed bin (35),

sun fixed bin init (1);

dc1 str1 char (64) var init ("Incremental Cost of RCRA Interim Status Standards"),

topstring (7) char (48) var ext static,

arg char (alen) unaligned based (aptr),

option char (15) var,

opstr char (100) var init ("vadd cap clos fyexp annop offe_dis tot_fyexp tot_ann_cost pctva"),

(str2, old_ind init (-), industry, ind_save) char (60) var,

filler char (3) var init (""),

zone char (3) init (""),

shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT"),

mo (12) char (10) var init ("January", "February", "March", "April", "May", "June", "July", "August", "September",

"October", "November", "December"),

days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");

dc1 (more_smt, repeat) bit (1) aligned init ("1'b");

dc1 (hlsu, industry_sw, order, up, ann_sw) bit (1) aligned init ("0'b");

dc1 (aptr, optr) pointer;

dc1 (substr, index, sum, length) builtin,

endpage condition,

(sysprint output, report output) file stream,

```

cu_sarg_ptr_ext entry (fixed bin, ptr, fixed bin, fixed bin (35));
dc1 decode_clock_value_time_entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
dc1 header_char (3), fixed bin (35));
dc1 annualize_npv_ext entry (char (64) var, file);
xpage;
on endpage (report) begin;
put skip (3) file (report) edit ("Sum in "Annual Closure Fund" represents year 1 of", clos_period, "year
s.")
(col (1), a, x (1), p"19", x (1), a);
put page file (report);
put file (report) edit (mo (month), dom, " ", year, hour, ":",
filler, trim (char (minute)), shift (sun), days (dow))
(col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
x (1), f (2), a (1), a, a, x (1), a, x (1), a (3));
put file (report) skip;
call header ("EXHIBIT REPORT", report);
put file (report) skip (3);
put file (report) skip edit
("Cost of Waste Management at the Generator Plant")
(col (38), a)
("Annual")
(col (89), a);
put file (report) skip edit
("First Year") (col (45), a)
("Annual") (col (71), a)
("Cost of") (col (88), a)
("Annual") (col (125), a);
put file (report) skip edit
("-----") (col (38), a)
("-----") (col (62), a)
("Off-Site") (col (88), a)
("Total") (col (103), a)
("Cost as a") (col (124), a);
if -industry_sw then put file (report) edit ("Industry") (col (5), a);
put file (report) edit ("-1978 Value") (col (27), a);
("Initial") (col (52), a)
("Closure") (col (64), a)
("Transportation") (col (85), a)
("First Year") (col (101), a)
("Total") (col (116), a)
("Percent of") (col (124), a);
put file (report) skip;
if industry_sw then put file (report) edit ("Industry") (col (5), a);
else put file (report) edit ("Segment") (col (5), a);
put file (report) edit ("Added") (col (29), a)
("Capital") (col (40), a)
("Expenses") (col (52), a)
("Fund") (col (66), a)
("Operating") (col (74), a)
("and Disposal") (col (86), a)
("Expenditure") (col (100), a)
("Annual Cost") (col (112), a)
("Value Added") (col (124), a);
put file (report) skip;
if industry_sw then put file (report) edit ("-----") (col (5), a);
else put file (report) edit ("-----") (col (5), a);
put file (report) edit ("-----") (col (27), a)
("-----") (col (40), a)

```

```

("-----") (col (52), a)
("-----") (col (64), a)
("-----") (col (74), a)
("-----") (col (85), a)
("-----") (col (100), a)
("-----") (col (112), a)
("-----") (col (124), a);

put file (report) skip (2);

end;
/* begin block */

```

```

arglp:
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to proceed;
else if arg = "-himpact" | arg = "-hl" then hlsw = "1'b";
else if arg = "-industry" | arg = "-ind" then industry_sw = "1'b";
else if arg = "-ann" then ann_sw = "1'b";
else if arg = "-sort" then do;
order = "1'b";
argno = argno + 1;
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then option = "pctva";
else if index (arg, "-") = 0 then do;
argno = argno - 1;
option = "pctva";
end;
else if index (opstr, arg) = 0 then do;
put skip edit ("Sort options are:", opstr) (a, x (2), a);
put skip (2);
return;
end;
else option = arg;
end;
else if arg = "-asc" | arg = "-desc" then do;
if "-order" then do;
put skip list ("If sort desired, list as argument.");
put skip list ("-Args -desc and -asc invalid without sort arg.");
put skip (2);
return;
end;
else if arg = "-asc" then up = "1'b";
end;
else do;
put skip list ("Args are: -himpact -industry");
put skip (2);
return;
end;
argno = argno + 1;
go to arglp;
/* end error do */

```

```

proceed:
if hlsw then hl_incr = num_hl_seg;
allocate rep_struct;
open file (report) title ("vfile_exhibit.report") print linesize (135) pagesize (57);

bits = clock ();
call decode_clock_value ($date_time
(bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code);
if hour = 12 then do;

```

```

if minute = 0 then sun = 3;
else sun = 2;
end;
if hour = 0 then w;
hour = 12;
if minute = 0 then sun = 4;
end;
if hour > 12 then do;
hour = hour - 12;
sun = 2;
end;
if minute < 10 then filler = "0";
topstring (7) = "(SMM)";

outputs (*, *, *, *) = 0.0;
do smt = 1 to num_segments + hl_incr;
rep_struct (smt).tva = 0e0;
rep_struct (smt).tcap = 0e0;
rep_struct (smt).tcicost = 0e0;
rep_struct (smt).tfye = 0e0;
rep_struct (smt).fy = 0e0;
rep_struct (smt).taoe = 0e0;
rep_struct (smt).tod = 0e0;
rep_struct (smt).ttac = 0e0;
do nsec = 1 to 5;
do freq = 1 to 4;
do l = 1 to 8;
do mod = 1 to num_models;
if smt <= num_segments then
outputs (freq, nsec, smt, l) =
state_reqs (nsec, l) - carry$ (smt, 1, mod, nsec, freq, l) / 1e6;
else outputs (freq, nsec, smt, l) =
outputs (freq, nsec, smt, l) +
state_reqs (nsec, l) + impacts$ (smt, 2, mod, nsec, freq, l) +
state_reqs (nsec, l) + impacts$ (smt, 1, mod, nsec, freq, l) / 1e6;
/* end mod */
end;
end; end; end;

tva, tcap, tfye, taoe, tod, ttac = 0e0;
gtcap, gtfye, gtaoe, gtod, gttac, gtv = 0e0;
smt = 1;
str2 = ind_struct$ (smt).ind_name;
industry = substr (str2, 1, index (str2, " ")-1);
ind_save = industry;
do while (more_smt);
counter = counter + 1;
if industry_sw then rep_struct (counter).name = industry;
else rep_struct (counter).name = str2;
do while (industry = ind_save & more_smt & repeat);
va = sum (ind_struct$ (smt).value_add (*));
cap = sum (outputs (3, *, smt, *));
num_gen = sum (stat_block$ (smt).size (*).gen_ons (*)) + sum (stat_block$ (smt).size (*).gen_offs (*));
if num_gen = 0 then alloc = 0;
else alloc = sum (stat_block$ (smt).size (*).gen_ons (*)) / num_gen;
fy = sum (outputs (1, *, smt, *));
cicost = sum (outputs (4, *, smt, *));
ann_clos = cicost / clos_period;
aoe = sum (outputs (2, *, smt, *));
aoe = aoe - outputs (2, 4, smt, 1);
/* not recurring tech */

```

```

od = outputs$ (2, 4, smt, 1);
if ann_sw then
  tac = (cap+fye)*.2716+aoe+od+clcost;
else tac = annualize_npv_ (fye, aoe+od, cap, clcost);
fye = fye + cap + ann_closta+od;
rep_struc (counter).tva = rep_struc (counter).tva + va;
rep_struc (counter).tcap = rep_struc (counter).tcap + cap;
rep_struc (counter).fclcost = rep_struc (counter).fclcost + ann_closta;
rep_struc (counter).fye = rep_struc (counter).fye + fye;
rep_struc (counter).tfye = rep_struc (counter).tfye + fye;
rep_struc (counter).taoe = rep_struc (counter).taoe + aoe;
rep_struc (counter).tod = rep_struc (counter).tod + od;
rep_struc (counter).ttac = rep_struc (counter).ttac + tac;
smt = smt + 1;
old_ind = str2;
if smt <= num_segments + hi_incr then do;
  str2 = ind_struc$ (smt).ind_name;
  industry = substr (str2, 1, index (str2, ":")-1);
end;
else more_smt = "0";
repeat = industry_sw;
end;
ind_save = industry;
repeat = "1";
end;
allocate list;
if order then call sort (option, up);
else do i = 1 to counter;
  list (i) = i;
end;
signal endpage (report);
do i = 1 to counter;
  if ~industry_sw then do;
    if i = 1 then industry = substr (rep_struc (list (i)).name, 1, index (rep_struc (list (i)).name, ":")-1);
  else do;
    str2 = substr (rep_struc (list (i)).name, 1, index (rep_struc (list (i)).name, ":")-1);
    if industry ~= str2 & ~order then do;
      signal endpage (report);
      industry = str2;
    end;
  end;
end;
if rep_struc (list (i)).tva = 0 then pctva = rep_struc (list (i)).ttac/rep_struc (list (i)).tva+1e2;
else pctva = 0;
if (industry_sw) then put file (report) skip edit
(rep_struc (list (i)).name) (col (1), a (25));
else put file (report) skip edit (substr
(rep_struc (list (i)).name, 1, index (rep_struc (list (i)).name, ":")))
(a)
(substr (rep_struc (list (i)).name, index (rep_struc (list (i)).name, ":") +2))
(skip, a (25));
put file (report) edit
(rep_struc (list (i)).tva) (col (27), p"zzz.z29v.99")
(rep_struc (list (i)).tcap) (col (41), p"z.z29v.99")
(rep_struc (list (i)).fye) (col (54), p"z.z29v.99")
(rep_struc (list (i)).fclcost) (col (65), p"z.z29v.99")
(rep_struc (list (i)).taoe) (col (76), p"z.z29v.99")
(rep_struc (list (i)).tod) (col (89), p"z.z29v.99")

```



```

(rep_struc (list (1).tfe) (col (102). p"z.z29v.99")
(rep_struc (list (1).ttac) (col (114). p"z.z29v.99")
(pctva) (col (127). t (7, 2));

end;

/* print summary page for all industries */

put file (report) skip (2);
if sum (rep_struc (*) .tva) = 0 then pctva = sum (rep_struc (*) .ttac)/sum (rep_struc (*) .tva)*1e2;
else pctva = 0;
put file (report) skip edit
("Total") (col (1). a)
(sum (rep_struc (*) .tva)) (col (27). p"zz.z29v.99")
(sum (rep_struc (*) .tcap)) (col (41). p"z.z29v.99")
(sum (rep_struc (*) .tfe)) (col (54). p"z.z29v.99")
(sum (rep_struc (*) .tccost)) (col (65). p"z.z29v.99")
(sum (rep_struc (*) .taoe)) (col (76). p"z.z29v.99")
(sum (rep_struc (*) .tod)) (col (89). p"z.z29v.99")
(sum (rep_struc (*) .tfe)) (col (102). p"z.z29v.99")
(sum (rep_struc (*) .ttac)) (col (114). p"z.z29v.99")
(pctva) (col (127). p"zz29v.99");

signal endpage (report);

close file (report);

free rep_struc;
free list;
topstring (7) = "";

Xpage;
sort;
dcl option char (15) var;
dcl up bit (1) aligned;
dcl item (counter) float dec;
dcl bound float dec;
dcl (i, j, k).fixed bin;
if option = "vadd" then do i = 1 to counter;
item (i) = rep_struc (i).tva;
end;
else if option = "cap" then do i = 1 to counter;
item (i) = rep_struc (i).tcap;
end;
else if option = "clos" then do i = 1 to counter;
item (i) = rep_struc (i).tccost;
end;
else if option = "fyexp" then do i = 1 to counter;
item (i) = rep_struc (i).tfe;
end;
else if option = "annop" then do i = 1 to counter;
item (i) = rep_struc (i).taoe;
end;
else if option = "offs_dis" then do i = 1 to counter;
item (i) = rep_struc (i).tod;
end;
else if option = "tot_ann_cost" then do i = 1 to counter;
item (i) = rep_struc (i).ttac;
end;
else if option = "pctva" then do i = 1 to counter;

```

```

if rep_struct (i).tva ^= 0 then
    item (i) = rep_struct (i).ttac/rep_struct (i).tva;
else item (i) = 0e0;
end;
bound = item (i);
k = 1;
do i = 1 to counter;
    do j = 1 to counter;
        if (up & item (j) < bound) | (~up & item (j) > bound) then do;
            bound = item (j);
            k = j;
        end;
    end;
    list (i) = k;
    if up then do;
        item (k) = 1000000000;
        bound = 1000000000;
    end;
    else do;
        item (k) = -1;
        bound = -1;
    end;
end;
end;
return;
end sort;
end exhibit;

```

```

/* ..... */
/* Written in 1979 at Arthur D. Little, Inc. */
/* Cambridge, Mass. */
/* ..... */
/* ..... */
fence_1 procedure (DISP, SIZE) returns (float dec);
/* ..... */
/* procedure to calculate and return cost of fencing */
dcl (DISP, disp) fixed bin,
    (SIZE, size) float dec,
    cost float dec;

dcl sqrt builtin;
Xpage;
disp = DISP;
size = SIZE;

go to code (disp);

code (1):
code (8):
code (9):
code (10):
code (5):
    cost = 325 + 48 * sqrt (127 * size**.65);
    return (cost);

code (2):
    cost = 325 + 48 * sqrt (60 * size);
    return (cost);

code (0):
code (3):
code (4):
    cost = 0;
    return (cost);

code (6):
code (7):
    cost = 325 + 48 * sqrt (4.356e4 + 1.37 * size**.9);
    return (cost);

end fence_1
/* sanitary landfill */
/* secure landfill */
/* dump */
/* drummed waste */
/* monofill */
/* size is in tonnes */

/* landfill */
/* size is in tonnes */

/* recycling */
/* incinerator */

/* disposal impoundment */
/* treat/store impoundment */
/* 1.37*size**.9 is in acres */

```

```
get_size_1 procedure (disp, tot_waste, size);
```

```
/* procedure to return size code for a given disposal method and volume */
```

```
dc1 (
```

```
    disp,  
    size) fixed bin;
```

```
dc1 tot_waste float dec (6);
```

```
go to code (disp);
```

```
code (1):
```

```
code (8):
```

```
code (9):
```

```
code (10):
```

```
    if tot_waste <= 1000 then size = 1;  
    else if tot_waste <= 2000 then size = 2;  
    else if tot_waste <= 5500 then size = 3;  
    else if tot_waste <= 20000 then size = 4;  
    else size = 5;  
    return;
```

```
code (2):
```

```
    size = 1;  
    return;
```

```
code (3):
```

```
code (4):
```

```
    if tot_waste <= 1000 then size = 1;  
    else if tot_waste <= 5000 then size = 2;  
    else if tot_waste <= 10000 then size = 3;  
    else if tot_waste <= 20000 then size = 4;  
    else size = 5;  
    return;
```

```
code (6):
```

```
code (7):
```

```
    if tot_waste <= .48 then size = 1;  
    else if tot_waste <= .79 then size = 2;  
    else if tot_waste <= 1.0 then size = 3;  
    else if tot_waste <= 1.7 then size = 4;  
    else if tot_waste <= 2.15 then size = 5;  
    else size = 6;  
    return;
```

```
code (0):
```

```
    size = 1;  
    return;
```

```
end get_size_;
```

```
/* sanitary landfill */  
/* secure landfill */  
/* dump */  
/* drummed waste */
```

```
/* landfill */
```

```
/* incinerator w/ scrubber */  
/* incinerator w/o scrubber */
```

```
/* surface impoundment; landfill closure */  
/* surface impoundment; dredge */
```

```
/* recycling */
```

```

/* .....
 *
 * Copyright (c) 1978 by Arthur D. Little, Inc.
 *
 * .....
 */

gtd: procedure;

Xinclude rcra;
dcl costs (5, 4, num_segments, nmodels, 8) based (costs_ptr) fixed dec (8, 3), /* 5-sect, (init, ann, cap, clos), segment, 8=a
ctivity */
    rep (num_segments+1-incr, nmodels, 5) based (rep_ptr) float dec, /* array for formal reporting of costs */
    (tprod (nmodels), tcosts (nmodels, 5)) float dec; /* for report totals */

dcl list (counter) based (list_ptr) fixed bin;
dcl list_ptr pointer;
dcl struc_ptr pointer;
dcl rep_ptr pointer;

dcl 1 rep_struc (num_segments+1-incr) aligned based (struc_ptr),
    2 name char (60),
    3 size (4),
    3 value_add float dec,
    3 tot_gen float dec,
    3 tot_trans float dec,
    3 tot_disp float dec,
    3 total float dec;

dcl aptr pointer, arg char (alen) unaligned based (aptr),
ec fixed bin (35), alen fixed bin;
dcl argno fixed bin init (1);

dcl (frec, l, j, k, ref, nsec, smt, num_smt, hl_incr init (0)) fixed bin,
discount float dec init (-.2716),
tpct float dec,
(gen init (1), trans init (2), tsdf init (3), tot init (4)) static fixed bin, /* for rep */
nmodels init (4) static fixed bin;
dcl counter fixed bin init (0);
dcl (month, dom, year, hour, minute, second, dow) fixed bin,
(bits, microsecond) fixed bin (71),
code fixed bin (35),
sun fixed bin init (1);

dcl va (4) float dec, (trep (4, 5), ttrep (4), tac) float dec (9);
dcl (tva, ttva) float dec;

dcl sname (4) char (8) var init (" Large", "Medium", " Small", " SWG"),
mtemp char (8) var init ("").

```



```

fname char (36) var;
option char (6) var;
str1 char (64) var init ("Incremental Costs of RCRA Interim Status Standards");
(str2, industry, ind_save) char (60) var;
str3 (2) char (20) var init ("by Segment", "by Industry");
opstr char (35) var init ("vadd gen trans disp total pctva");
filler char (3) var init (" ");
zone char (3) init (" ");
shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT");
mo (12) char (10) var init ("January", "February", "March", "April", "May", "June", "July", "August",
"September", "October", "November", "December");
days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");

dcl (did_rep, sm_on_line, nostate, industry_sw, model_sw, order, up, hlew) bit (1) aligned init ("0'b");
dcl (more_smt, repeat) bit (1) aligned init ("1'b");

dcl (costs_ptr) pointer;

dcl (sum, index, char, clock, length, ltrim, substr) builtin;
dcl cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35));
dcl decode_clock_value_sdate_time entry (fixed bin (71), fixed bin (71), fixed bin, fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin (71), fixed bin,
char (3), fixed bin (35));
dcl header_ext entry (char (84) var, file);
dcl (sysprint output, report output, sysin input) file stream;
dcl endpage condition;

xpage:
on endpage (report) begin;
put page file (report) edit (mo (month), dom, " ", year, hour, ":",
filler, ltrim (char (minute)), shift (sun), days (dow))
(col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
x (1), f (2), a (1), a, x (1), a, x (1), a (3));
put file (report) skip;
if industry_sw then k = 2;
else k = 1;
call header_ ("GTD Report" || str3 (k), report);
put file (report) skip (2);
if industry_sw then put file (report) edit ("Industry") (col (4), a);
else put file (report) edit ("Segment") (col (4), a);
put file (report) edit ("Value Added") (col (31), a)
("Generation") (col (48), a)
("Transportation") (col (66), a)
("Disposal") (col (87), a)
("TOTAL") (col (108), a)
("Xvadd") (col (126), a);
put file (report) skip;
if industry_sw then put file (report) edit ("-----") (col (4), a);
else put file (report) edit ("-----") (col (4), a);
put file (report) edit ("-----") (col (31), a)
("-----") (col (48), a)
("-----") (col (66), a)
("-----") (col (87), a)
("-----") (col (108), a)
("-----") (col (126), a);
put file (report) skip (3);
end;

xpage:
arglp:
/* end begin block */

```

```

call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to start;

if arg = "-nostate" | arg = "-ns" then nostate = "i'b";
else if arg = "--industry" | arg = "--ind" then industry_sw = "i'b";
else if arg = "-hi" then hls = "i'b";
else if arg = "-model" | arg = "-md" then do;
  model_sw = "i'b";
  atemp = "Totals";
end;
else if arg = "-sm" then do;
  sm_on_line = "i'b";
  argno = argno + 1;
  call cu_sarg_ptr (argno, aptr, alen, ec);
  if ec = 0 then fname = "gtd.report";
  else if index (arg, "--") = 0 then do;
    argno = argno + 1;
    fname = "gtd.report";
  end;
  else fname = arg;
end;
else if arg = "-sort" then do;
  order = "i'b";
  argno = argno + 1;
  call cu_sarg_ptr (argno, aptr, alen, ec);
  if ec = 0 then option = "pctva";
  else if index (arg, "--") = 0 then do;
    argno = argno + 1;
    option = "pctva";
  end;
  else if index (opstr, arg) = 0 then do;
    put skip edit ("Sort options are:", opstr) (a, x (2), a);
    put skip (2);
    return;
  end;
  else option = arg;
end;
else if arg = "-asc" | arg = "-desc" then do;
  if "order" then do;
    put skip list ("If sort desired, list as argument.");
    put skip list ("Args -desc and -asc invalid without sort arg.");
    return;
  end;
  else if arg = "-asc" then up = "i'b";
end;
else do;
  put skip list
  ("Args are: -nostate -sm <name> -industry -model -sort <option> -desc -asc -hi");
  put skip (2);
  return;
end;
argno = argno + 1;
go to arglp;

Xpage;
start;

allocate costs;

if hls then hi_incr = num_hi_seq;

```

```

allocate rep_struct;
allocate rep;
if sm_on_line then fname = "gtd.report";
open file (report) title ("vfile_ "||fname) print linesize (135);

bits = clock ();
call decode_clock_value$(date,time
(bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code));
if hour = 12 then do;
if minute = 0 then sun = 3;
else sun = 2;
end;
else if hour = 0 then do;
hour = 12;
if minute = 0 then sun = 4;
end;
else if hour > 12 then do;
hour = hour - 12;
sun = 2;
end;
if minute < 10 then filler = "0";

num_smt = num_segments;
if nostate then state_regs (+, *) = 1; /* no exclusion option */
do smt = 1 to num_smt;
do j = 1 to nmodels;
rep_struct(smt).size(j).value_add = 0e0;
rep_struct(smt).size(j).tot_gen = 0e0;
rep_struct(smt).size(j).tot_trans = 0e0;
rep_struct(smt).size(j).tot_disp = 0e0;
rep_struct(smt).size(j).total = 0e0;
do nsec = 1 to 5;
do freq = 1 to 4;
do i = 1 to 8;
costs(nsec, freq, smt, j, i) =
(state_regs(nsec, i)
+ carry$(smt, 1, j, nsec, freq, i)) * 1e-6;
end; end; end; end; end;

do smt = 1 to num_smt;
do j = 1 to nmodels;
rep(smt, j, gen) = sum(costs(1, 2, smt, j, *)) + sum(costs(2, 2, smt, j, *));
rep(smt, j, gen) =
rep(smt, j, gen) + (sum(costs(1, 1, smt, j, *)) + sum(costs(2, 1, smt, j, *))) * discount;
/* 3003 */
rep(smt, j, trans) = sum(costs(3, 2, smt, j, *)) + sum(costs(3, 2, smt, j, *)) * discount;
/* 3004, 3005 */
rep(smt, j, tsdf) = sum(costs(4, 2, smt, j, *)) + sum(costs(5, 2, smt, j, *));
rep(smt, j, tsdf) =
rep(smt, j, tsdf) + sum(costs(4, 1, smt, j, *)) + sum(costs(5, 1, smt, j, *)) * discount;
rep(smt, j, tsdf) = rep(smt, j, tsdf) + sum(costs(4, 3, smt, j, *)) * discount;
rep(smt, j, tot) = rep(smt, j, gen) + rep(smt, j, tsdf) + rep(smt, j, trans);
rep(smt, j, 5) = 0;
if ind_struct$(smt).value_add(j) > 0 then
rep(smt, j, 5) = rep(smt, j, tot) / ind_struct$(smt).value_add(j) * 1e2;
end;

```

```

end;
else rep (smt, j, 5) = 0;
end;

/* end j */
/* end smt loop */

/* check to see if rep was initialized */
did_rep = "1";
if hisw then do;
  do smt = 1 to hi_incr;
    do nsec = 1 to 5;
      do freq = 1 to 4;
        do i = 1 to 8;
          costs (nsec, freq, smt, i, i) =
            (state_regs (nsec, i)
            + impacts$ (num_segments + smt, 1, 1, nsec, freq, i)
            + impacts$ (num_segments + smt, 2, 1, nsec, freq, i)) * 1e-6;
          do j = 1 to nmodels;
            rep_struct (num_segments+smt).size (j).value_add = 0e0;
            rep_struct (num_segments+smt).size (j).tot_gen = 0e0;
            rep_struct (num_segments+smt).size (j).tot_trans = 0e0;
            rep_struct (num_segments+smt).size (j).tot_disp = 0e0;
            rep_struct (num_segments+smt).size (j).total = 0e0;
          end;
        end;
      end;
    end;
  end;
end;

do smt = 1 to hi_incr;
  do j = 1 to nmodels;
    rep (num_segments+smt, j, gen) = 0;
    rep (num_segments+smt, j, trans) = 0;
    rep (num_segments+smt, j, tsdf) = 0;
    rep (num_segments+smt, j, tot) = 0;
    rep (num_segments+smt, j, 5) = 0;
  end;

  rep (num_segments+smt, 1, gen) = sum (costs (1, 2, smt, 1, *)) + sum (costs (2, 2, smt, 1, *));
  rep (num_segments+smt, 1, gen) =
    rep (num_segments+smt, 1, gen) + (sum (costs (1, 1, smt, 1, *)) + sum (costs (2, 1, smt, 1, *))) * di;

  /* 3003 */
  rep (num_segments+smt, 1, trans) = sum (costs (3, 2, smt, 1, *)) + sum (costs (3, 2, smt, 1, *)) * disco;

  /* 3004, 3005 */
  rep (num_segments+smt, 1, tsdf) = sum (costs (4, 2, smt, 1, *)) + sum (costs (5, 2, smt, 1, *));
  rep (num_segments+smt, 1, tsdf) = rep (num_segments+smt, 1, tsdf) + sum (costs (4, 4, smt, 1, *));
  rep (num_segments+smt, 1, tsdf) =
    rep (num_segments+smt, 1, tsdf) + (sum (costs (4, 1, smt, 1, *)) + sum (costs (5, 1, smt, 1, *))) * di;

  rep (num_segments+smt, 1, tsdf) = rep (num_segments+smt, 1, tsdf) + sum (costs (4, 3, smt, 1, *)) * disco;
  rep (num_segments+smt, 1, tot) = rep (num_segments+smt, 1, gen) + rep (num_segments+smt, 1, tsdf)
    + rep (num_segments+smt, 1, trans);
  rep (num_segments+smt, 1, 5) = 0;
  if ind_struct$ (num_segments+smt).value_add (1) > 0 then
    rep (num_segments+smt, 1, 5) = rep (num_segments+smt, 1, tot) / ind_struct$ (num_segments+smt).value;

  /* add (1) * 1e2;
  end;
end;

signal endpage (report);
tprod (+) = 0; tcosts (+, *) = 0;
tva, ttva, va (+), ttrp (+, *), ttprp (+), tpct, tao = 0;
smt = 1;

```

```

str2 = ind_struct$ (smt).ind_name;
industry = substr (str2, 1, index (str2, ":") - 1);
ind_save = industry;
do while (more_smt);
  counter = counter + 1;
  if industry_sw then rep_struct (counter).name = industry;
  else rep_struct (counter).name = str2;
  do while (industry = ind_save & more_smt & repeat);
    do j = 1 to nmodels;
      rep_struct (counter).size (j).value_add = rep_struct (counter).size (j).value_add + ind_struct$ (smt
        ).value_add (j);
      rep_struct (counter).size (j).tot_gen =
        rep_struct (counter).size (j).tot_gen + rep (smt, j, 1);
      rep_struct (counter).size (j).tot_trans =
        rep_struct (counter).size (j).tot_trans + rep (smt, j, 2);
      rep_struct (counter).size (j).tot_disp =
        rep_struct (counter).size (j).tot_disp + rep (smt, j, 3);
      rep_struct (counter).size (j).total =
        rep_struct (counter).size (j).total + rep (smt, j, 4);
    end;
    do j = 1 to nmodels;
      tprod (j) = tprod (j) + ind_struct$ (smt).value_add (j);
      do k = 1 to 5;
        tcosts (j, k) = tcosts (j, k) + rep (smt, j, k);
      end;
    end;
    smt = smt + 1;
    if smt <= num_segments + hi_incr then do;
      str2 = ind_struct$ (smt).ind_name;
      industry = substr (str2, 1, index (str2, ":") - 1);
    end;
    else more_smt = "0";
    repeat = industry_sw;
  end;
  repeat = "1";
  ind_save = industry;
end;

allocate list;
if order then call sort (option, up);
else do i = 1 to counter;
  list (i) = i;
end;

do i = 1 to counter;
  ref = list (i);
  if (industry_sw & ref > num_segments) then put file (report) edit (rep_struct (ref).name) (x (2), a);
  else put file (report) edit (substr
    (rep_struct (ref).name, 1, index (rep_struct (ref).name, ":")))
    (x (2), a)
    (substr (rep_struct (ref).name, index (rep_struct (ref).name, ":") + 2))
    (skip, x (3), a);
  if model_sw then do j = 1 to nmodels;
    if ref <= num_segments then do;
      if rep_struct (ref).size (j).value_add = 0 then
        tpct = rep_struct (ref).size (j).total /
          rep_struct (ref).size (j).value_add * 1e2;
      else tpct = 0e0;
      put file (report) skip edit
        (mname (j)) (x (10), a)
        (rep_struct (ref).size (j).value_add) (col (32), f (7, 1))
        (rep_struct (ref).size (j).tot_gen) (col (48), f (9, 3))

```



```

(rep_struct(ref).size(j).tot_trans) (col (68), f (9, 3))
(rep_struct(ref).size(j).tot_disp) (col (65), f (9, 3))
(tpct) (col (124), f (7, 2));
/* end if do */
end;
if sum (rep_struct(ref).size(*)-value_add) /= 0 then
  tpct = sum (rep_struct(ref).size(*)-total)/
  sum (rep_struct(ref).size(*)-value_add) * 1e2;
else tpct = 0e0;
put file (report) skip
  edit (mtemp) (x (10), a)
  (sum (rep_struct(ref).size(*)-value_add)) (col (32), f (7, 1))
  (sum (rep_struct(ref).size(*)-tot_gen)) (col (48), f (9, 3))
  (sum (rep_struct(ref).size(*)-tot_trans)) (col (68), f (9, 3))
  (sum (rep_struct(ref).size(*)-tot_disp)) (col (85), f (9, 3))
  (sum (rep_struct(ref).size(*)-total)) (col (104), f (9, 3))
  (tpct) (col (124), f (7, 2));
put file (report) skip (1);
end;
put file (report) skip (2);
put file (report) edit ("Total of All Industries") (a);
put file (report) skip;
if model_sw then do j = 1 to nmodels;
  if tprod(j) = 0 then tpct = 0;
  else tpct = tcosts(j, 4)/tprod(j)*1e2;
  put file (report) skip edit
    (aname(j), tprod(j), (tcosts(j, k) do k = 1 to 4), tpct)
    (x (10), a, col (29), f (10, 1), x (4), (4) (f (10, 2), x (9)), x (3), f (7, 2));
end;

end;
if sum (tprod(*)) = 0 then tpct = 0;
else tpct = sum (tcosts(*, 4))/sum (tprod(*))*1e2;
put file (report) skip edit
  ("TOTAL", sum (tprod(*)), (sum (tcosts(*, k) do k = 1 to 4), tpct)
  (x (10), a, col (29), f (10, 1), x (4), (4) (f (10, 2), x (9)), x (3), f (7, 2));
put file (report) skip;
close file (report);
free rep; free rep_struct;
free costs;
free list;

Xpage;
sort;
dcl option char (8) var;
dcl up bit (1) aligned;
dcl item (counter) float dec (9);
dcl bound float dec (9);
dcl (i, j, k) fixed bin;
  if option = "vadd" then do i = 1 to counter;
    item (i) = sum (rep_struct(i).size(*)-value_add);
  end;
  else if option = "total" then do i = 1 to counter;
    item (i) = sum (rep_struct(i).size(*)-total);
  end;
  else if option = "gen" then do i = 1 to counter;
    item (i) = sum (rep_struct(i).size(*)-tot_gen);
  end;
  else if option = "trans" then do i = 1 to counter;
    item (i) = sum (rep_struct(i).size(*)-tot_trans);
  end;

```

```

end;
else if option = "disp" then do i = 1 to counter;
  item (i) = sum (rep_struct (i).size (*) .tot_disp);
end;
else if option = "pctva" then do i = 1 to counter;
  if sum (rep_struct (i).size (*) .value_add) ^= 0 then
    item (i) = sum (rep_struct (i).size (*) .total) /
      sum (rep_struct (i).size (*) .value_add) * 1e2;
  else item (i) = 0e0;
end;
bound = item (i);
k = 1;
do i = 1 to counter;
  do j = 1 to counter;
    if (up & item (j) < bound) { (~up & item (j) > bound) then do;
      bound = item (j);
      k = j;
    end;
  end;
  list (i) = k;
  if up then do;
    item (k) = 100000;
    bound = 100000;
  end;
  else do;
    item (k) = -1;
    bound = -1;
  end;
end;
end;
return;
end sort;
end gtd;

```

header_.pl1

06/04/81 0141.1r w 08/04/81 0046.5

5382

```
header_: procedure (report_name, outfile);
  dcl report_name char (64) var;
  dcl outfile file;
  dcl topstring (7) char (48) var ext static;
  dcl i fixed bin;
  dcl length builtin;

  do i = 1 to 2;
    if topstring (i) ^= "" then
      put file (outfile) skip edit (topstring (i)) (col (65 - length (topstring (i))/2), a);
    end;
    put file (outfile) skip edit (report_name) (col (65 - length (report_name)/2), a);
    do i = 3 to 7;
      if topstring (i) ^= "" then
        put file (outfile) skip edit (topstring (i)) (col (65 - length (topstring (i))/2), a);
      end;
      put file (outfile) skip;
    end header_;
```

```

/* ..... */
/*
/* Written in 1979 at Arthur D. Little, Inc.
/* Cambridge, Mass. 02140
/* ..... */
/* ..... */
/* coded 7-15-80 by K. A. Speelman
/* changed 8-15-80 by TJM for annualized npv
/* ..... */

hl_output: procedure;

/* program to print cost reports for high impact industries */

dcl (i, j, k, l, m, fs, displsw,
     alien, argno init (1)) fixed bin,
     ec fixed bin (35),
     (num_displsw init (17), num_sizes, num_disch, num_alloc_models) fixed bin,
     (ttemp (report_structs.num_models), costs_five (report_structs.num_models)) float dec (6),
     (sc1, sc2, sca, stw, stp, anpv, sum_costs_five, sum_ttemp) float dec;

dcl temp float dec (12);
dcl scaled_costs float dec (12);

/* used for scale factor adjustment */
/* 1-land,2-bldg,3-eqp,4-oda,5-ann nond,6-init,7-init nond */

dcl (pa_ptr, splr, ts_ptr) pointer;

dcl (month, dow, year, hour, minute, second, dow) fixed bin,
     (bits, microsecond) fixed bin (71),
     code fixed bin (35),
     sun fixed bin init (1),
     filler char (3) var init (""),
     zone char (3) init (""),
     shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT"),
     mo (12) char (10) var init ("January", "February", "March", "April", "May", "June",
     "July", "August", "September", "October", "November", "December"),
     days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");

dcl (seg_name, temp_name) char (64) var,
     report_title char (64) varying;

dcl arg char (alen) unaligned based (aptr), out_name char (36) var init (""), test_code char (1) var init ("");

dcl rd_str char (5) init (":READ"), star char (2) init (".."),
     fname (2) char (7) init ("RCD040", "RCD040"), ft (2) char (2) init (".."),
     ftype (12) char (8) var init ("WOOLSCR", "WOOLFIN", "WOVSLIN", "WOVCPX", "WOVDES",
     "KNITSPL", "KNITCPX", "HOSIERY", "CARPET", "YARN", "NONWOVEN", "FELT"),
     oname char (64) varying init ("WOOL SCOURING: S DIRECT"), /* textile mode */
     disp_str (17) char (32) var init ("ONSITE SAN LANDFILL", "OFFSITE SAN LANDFILL", "ONSITE DUMP", "ONSITE PILE",
     "ONSITE DISP SI", "NO WASTE", "ONSITE LANDFARM", "OFFSITE LANDFARM", "ONSITE INCIN NO SCRUB",
     "OFFSITE INCIN NO SCRUB", "OFFSITE POTW", "RECYCLE", "OFFSITE SEC LANDFILL", "ONSITE SEC LANDFILL",
     "ONSITE INCIN SCRUB", "OFFSITE INCIN SCRUB", "RECYCLE"),
     (size_sw init (0), model_sw init (1), dir_sw) fixed bin;

dcl check bit (1) aligned init ("1b"),
     (cards, totals, mpr) bit (1) aligned init ("0b").

```

```

dpra_str char (9),
  v1 char (7) init ("vfile_"),
  state_str char (48) var init ("With State Regulations"),
  iss_str char (48) var init ("Interim Status Standards"),
  names (5) char (64) var init
    ("Textiles", "Leather Tanning&Finishing",
     "Non-Ferrous Metals",
     "Organic Chemicals", "Inorganic Chemicals");

dcl 1 report_struct aligned ext static,
  2 ind char (64) var,
  2 num_models fixed bin,
  2 state bit (1),
  2 slw bit (1),
  2 nowd bit (1),
  2 iss bit (1),
  2 mod (68),
  3 mp_name char (64) var,
  3 scale_factor float dec (12),
  3 num_sub_cat fixed bin,
  3 sub (12),
  4 scname char (64) var,
  4 num_codes fixed bin,
  4 code (7) fixed bin,
  4 num_plants fixed bin,
  4 costs (7) float dec (12),
  4 tot_type_waste float dec (12),
  4 dispw fixed bin;

dcl 1 total_struct (report_struct.num_models) aligned based (ts_ptr), /* for totals report */
  2 plants fixed bin,
  2 waste float dec (6),
  2 cost (2) float dec (6);

dcl (sysprint, dpra_file, out_file) file stream output,
  (substr, index, clock, fixed, char, ltrim, length) builtin,
  cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35)),
  decode_clock_value$state_time entry (fixed bin (71), fixed bin,
  fixed bin, fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
  fixed bin, char (3), fixed bin (35)),
  annualize_npv_ext entry (float dec, float dec, float dec, float dec) returns (float dec);

%page;
argdp:
  call cu_sarg_ptr (argno, aptr, alen, ec);
  if ec = 0 then go to start;

  /* got all arguments */
  else if arg = "-card" | arg = "-cd" then cards = "1'b";
  else if arg = "-totals" | arg = "-tt" then totals = "1'b";
  else if arg = "-mpr" then mpr = "1'b";
  else if arg = "-sm" then do;
    argno = argno + 1;
    call cu_sarg_ptr (argno, aptr, alen, ec);
    if ec = 0 then out_name = "hreport.list";
    else if index (arg, "-") = 0 then do;
      argno = argno - 1;
      out_name = "hreport.list";
    end;
  else out_name = arg;
end;

/* -sm <out_name> case */
/* end arg do group */

```



```

put skip list ("Args are: -card -total -mpr -sm <name>");
put skip (2);
return;
/* return from hl_output */
/* end error do */

end;
argno = argno + 1;
go to arglp;

start:

if cards then open file (dpra_file) title (v||"dpra.report") print linesize (135);
if -(mpritotals) then totals = "0'b;
ltemp (+) = 0e0; costs_five (+) = 0e0;
sum_costs_five = 0e0; sum_ttemp = 0e0;
if out_name = "" then open file (out_file) title ("syn_user_output") print linesize (135);
else open file (out_file) title (v||out_name) print linesize (135);

if report_struct$.slwv~report_struct$.nowd
then report_title = "Waste Disposal and Surface Impoundment Costs";
else if report_struct$.slw then report_title = "Surface Impoundment Costs";
else report_title = "Waste Disposal Costs";

temp_name = substr (report_struct$.ind, 1, index (report_struct$.ind, "-")-1);
if temp_name = names (1) then do;
  seq_name = names (1);
  dpra_str = "text.dpra";
  num_sizes = 4; num_disch = 2; num_alloc_models = 12; /* constants for allocation of space */
  if cards then do;
    end;
  end;
else if temp_name = names (2) then do;
  seq_name = names (2);
  dpra_str = "leth.dpra";
  fname (1) = "RCRA040";
  ft (1) = "";
  ftype (1) = "CHROME S"; ftype (2) = "NONCHRM"; ftype (3) = "RETAN";
  ftype (4) = "NO BEAM"; ftype (5) = "BLUE"; ftype (6) = "SHEARL";
  ftype (7) = "FINISH"; ftype (8) = "CHROME P";
  num_sizes = 5; num_disch = 1; num_alloc_models = 8;
  if cards then do;
    end;
  end;
else if temp_name = names (3) then do;
  seq_name = names (3);
  if temp_name = names (3) then do;
    end;
  else if temp_name = names (4) then do;
    seq_name = names (4);
    end;
  else if temp_name = names (5) then do;
    seq_name = names (5);
    end;
  end;
end;

if mpritotals then do;
  bits = clock ();
  call decode_clock_value_sdate_time
    (bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code);
  if hour = 12 then do;
    if minute = 0 then sun = 3;
    else sun = 2;
  end;
  else if hour = 0 then do;

```

```

hour = 12;
if minute = 0 then sun = 4;
end;
else
if hour > 12 then do;
hour = hour - 12;
sun = 2;
end;
if minute < 10 then filler = "0";
put file (out_file) edit (mo (month), dom, ".", year, hour, "-");
filler, trim (char (minute)), shift (sum), days (dow)
(col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
x (1), f (2), a (1), a, x (1), a, x (1), a (3));
put file (out_file) skip;
put file (out_file) edit (seq_name) (col (50-length (seq_name)/2), a);
put file (out_file) skip;
if report_structs.iss then do;
put file (out_file) edit (iss_str) (col (50-length (iss_str)/2), a);
put file (out_file) skip;
end;
if report_structs.state then do;
put file (out_file) edit (state_str) (col (50-length (state_str)/2), a);
put file (out_file) skip;
end;
put file (out_file) edit (report_title) (col (50-length (report_title)/2), a);
put file (out_file) skip (3);
if mpr then do;
put file (out_file) edit ("Land", "Bldg", "Equip", "Non-d Rec", "Initial", "Non-d Init")
(col (15), a, col (26), a, col (37), a, col (50), a, col (60), a, col (73), a, col (86), a);
put file (out_file) skip (2);
end;
if totals then do;
allocate total_struct;
total_struct (*).cost (*) = 0;
total_struct (*).waste = 0;
total_struct (*).plants = 0;
end;
end;
end;
end;

/* set up totals report */
/* end mpr */
/* end totals set up */
/* end header do */

xpage;
/* main loop of program */
do i = 1 to report_structs.num_models;
if temp_name = names (i) & cards then do;
if check then if index (report_structs.mod (i).mp_name, "DIRECT") > 0 then dir_sw = 1;
else do;
dir_sw = 2;
model_sw = 0;
check = "0-b";
end;
if substr (report_structs.mod (i).mp_name, 1, index (oname, "-")-1)
then size_sw = size_sw + 1;
else do;
model_sw = model_sw + 1;
oname = report_structs.mod (i).mp_name;
size_sw = 1;
end;
end;
else if temp_name = names (2) & cards then do; /* parse model plants for temp_name = names(2) */
dir_sw = 1;
/* no indirect */
/* else group */
/* textile do group */
/* parse model plants for temp_name = names(2) */

```

```

size_sw = size_sw + 1;
if model_sw = 1 & size_sw = 6 then do;
  model_sw = model_sw + 1;
  size_sw = 1;
end;
else if size_sw = 4 & (model_sw = 2 | model_sw = 4) then do;
  model_sw = model_sw + 1; size_sw = 1;
end;
else if size_sw = 3 & (model_sw = 3 | model_sw = 5 | model_sw = 6 | model_sw = 7) then do;
  model_sw = model_sw + 1; size_sw = 1;
end;
do j = 1 to report_structs.mod (1).num_sub_cat; /* leather do group */
  dispw = report_structs.mod (1).sub (j).dispsw;
end;

/* scale O&M for total report */
scaled_costs = report_structs.mod (1).sub (j).costs (4) *
report_structs.mod (1).scale_factor; /* print reports, based on totals and mpr switches */
temp = report_structs.mod (1).sub (j).costs (5);
report_structs.mod (1).sub (j).costs (5) =
  annualize_npv_ ((temp), 0, 0, 0);

if cards then do;
  put file (dpra_file) edit
  (report_structs.mod (1).mp_name||" "||report_structs.mod (1).sub (j).scname) (col (2), a);
  put file (dpra_file) skip edit (trim (char (report_structs.mod (1).sub (j).num_plants))) (col (2)
  , a)
  ((fixed (report_structs.mod (1).sub (j).costs (1), 10) do 1 = 1 to 7))
  (col (10), 7) (r (10), x (1)));
  put file (dpra_file) skip edit (report_structs.mod (1).sub (j).num_codes) (col (2), f (2))
  ((report_structs.mod (1).sub (j).code (1) do 1 = 1 to report_structs.mod (1).sub (j).num_codes
  ))
  (col (8), (report_structs.mod (1).sub (j).num_codes) (r (3), x (3)));
end;
if mpr then do;
  put file (out_file) list (report_structs.mod (1).mp_name||" "||report_structs.mod (1).sub (j).scn
  , a)
  ("||trim (char (report_structs.mod (1).sub (j).num_plants))||");
  put file (out_file) skip;
  put file (out_file) edit ((fixed (report_structs.mod (1).sub (j).costs (1), 10) do 1 = 1 to 7))
  (col (10), 7) (r (10), x (1)));
  put file (out_file) skip (2);
end;
if totals then do;
  total_struct (1).cost (1) = total_struct (1).cost (1) +
  report_structs.mod (1).sub (j).num_plants *
  report_structs.mod (1).sub (j).costs (1) +
  report_structs.mod (1).sub (j).costs (2) +
  report_structs.mod (1).sub (j).costs (3) +
  report_structs.mod (1).sub (j).costs (6) +
  report_structs.mod (1).sub (j).costs (7);
  total_struct (1).cost (2) = total_struct (1).cost (2) + report_structs.mod (1).sub (j).num_plants *
  (scaled_costs);
  costs_five (1) = costs_five (1) +
  report_structs.mod (1).sub (j).costs (5) *
  report_structs.mod (1).sub (j).num_plants;
  ttemp (1) = ttemp (1) + temp * report_structs.mod (1).sub (j).num_plants;
  total_struct (1).waste = total_struct (1).waste + report_structs.mod (1).sub (j).tot_type_waste *

```

```

        report_struct.mod (i).sub (j).num_plants;
        total_struct (i).plants = total_struct (i).sub (j).num_plants;
        /* end total report do group */
        /* end j */
        /* dpra cards do group */
    end;
end;
close file (dpra_file);
end hl_output;

```

hldata_report: procedure;

```

dcl (argno init (1), alen, sic, nplants, num_sub_cat, x,
     nws, ntrg, num_reg_plants, num_nreg_plants, i, j, k, l, m, n, tech, np_trg, num_req, reqs,
     num_size) fixed bin;
dcl (volume, num_di_pp, sa, num_sl_pp, num, size, scale) float dec,
     (prod pic'z.zzz.z9', vadd pic'$.99v.99');
dcl (fn, ind_name, date, mname, sname, wname) char (84) var;
dcl filestr char (20) var init ('');
dcl alist char (20) init ('lethtextorgainornonf');
dcl arg char (alen) unaligned based (aptr);
dcl ec fixed bin (35);
dcl aptr pointer;
dcl true bit (1) aligned init ('i-b');
dcl si_present bit (1) aligned;
dcl (length, substr) builtin;
dcl endfile condition;
dcl datafile file ext;
dcl (sysprint output, report print output) file stream;
dcl cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35));

```

```

on endfile (datafile) begin;
  close file (datafile);
  if length (filestr) > 4 then filestr = substr (filestr, 5);
  else filestr = '';
  go to start;
end;

```

%page;
arglp;

```

call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then do;
  if filestr = '' then do;
    put skip list ('Args are: -<Industry> -all');
    put skip list ('An argument must be specified');
    return;
  end;
  else go to start;
end;
if arg = '-leather' | arg = '-leth' then filestr = filestr || '-leth';
else if arg = '-textile' | arg = '-text' then filestr = filestr || '-text';
else if arg = '-organic' | arg = '-orga' then filestr = filestr || '-orga';
else if arg = '-inorganic' | arg = '-inor' then filestr = filestr || '-inor';
else if arg = '-nonferrous' | arg = '-nonf' then filestr = filestr || '-nonf';
else if arg = '-all' then filestr = alist;
else do;
  put skip list ('Args are : -<Industry> -all');
  put skip (2);
end;
argno = argno + 1;

```



```

start:
go to arglp;

do while (filestr ~ " ");
  fn = substr (filestr, 1, 4) || ".hldata";
  open file (datafile) title ("vfile_";fn);
  open file (report) title ("vfile_hldata-report") print linesize (135);
  get file (datafile) list (ind_name, date, sic);
  put file (report) page;
  (date) (skip, col (65 - length (ind_name)/2), a)
  (sic) (skip, col (63), f (4));

  put file (report) skip (2);
  do while (true);
    get file (datafile) list (wname, sic, prod, vadd, nplants, scale, num_sub_cat);
    put file (report) edit ("Model:", wname) (col (5), a, col (13), a)
    ("-----") (skip, col (5), a);
    put file (report) skip;
    put file (report) edit ("SIC Code:") (col (7), a)
    (sic) (col (21), f (4))
    ("Plant Production(MT):") (skip, col (7), a)
    (prod) (col (31), p"z.zzz.zz")
    ("Segment Value Added($M):") (skip, col (7), a)
    (vadd) (col (32), p"$,$$v.99")
    ("Scale Factor:") (skip, col (7), a)
    (scale) (col (30), f (10, 6))
    ("Number of Plants:") (skip, col (7), a)
    (nplants) (col (29), f (4));
    put file (report) skip (3);
    do j = 1 to num_sub_cat;
      get file (datafile) list (scname, num_req_plants, num_reg_plants, nus, ntrg, sl_present);
      put file (report) edit ("Subcategory:", scname) (col (10), a, col (24), a);
      put file (report) skip;
      put file (report) edit ("Number of Plants:(non req)", num_req_plants) (col (12), a, col (41), f
      put file (report) edit ("Number of Plants:(reg)", num_reg_plants) (col (12), a, col (41), f (3));
      put file (report) skip (3);
      ("-----") (skip, col (15), a);
      put file (report) skip;
      put file (report) edit ("Waste Name") (col (25), a)
      ("Volume (MT/yr)") (col (46), a)
      ("Waste Tech.") (col (63), a)
      ("Num. Disp Imp") (col (76), a)
      ("Surface Area") (col (92), a);
      put file (report) skip edit ("-----") (col (25), a)
      ("-----") (col (46), a)
      ("-----") (col (63), a)
      ("-----") (col (76), a)
      ("-----") (col (92), a);
      do k = 1 to nus;
        get file (datafile) list (wname, volume, tech, num_d1_pp, sa);
        put file (report) edit (wname) (col (20), a (20))
        (volume) (col (48), f (9))
        (tech) (col (68), f (3))
        (num_d1_pp) (col (80), p"zz.zzzv.zz")
        (sa) (col (95), p"zz.zzzv.zz");
      end;
    end;
  end;

```

```

put file (report) skip (5);
put file (report) edit ("Technical Requirement Info") (col (15), a)
("-----") (col (15), a);

put file (report) skip;
put file (report) edit ("Group Number") (col (18), a)
("Number of Plants") (col (32), a)
("Requirements") (col (50), a);

put file (report) edit ("-----") (col (18), a)
("-----") (col (32), a)
("-----") (col (50), a);

do i = 1 to ntrq;
  get file (datafile) list (np_trq, num_req);
  put file (report) edit (i) (col (23), f (2))
  (np_trq) (col (39), f (3));
  do n = 1 to num_req;
    get file (datafile) list (reqs);
    put file (report) edit (reqs) (col (53), f (3));
    put file (report) skip;
  end;
end;

put file (report) skip (5);
if sl_present then do;
  put file (report) edit ("Treatment/Storage Impoundment Info") (col (15), a)
  ("-----") (col (15), a);

  put file (report) skip;
  put file (report) edit ("Number") (col (20), a)
  ("Size") (col (30), a);
  put file (report) edit ("-----") (col (20), a)
  ("-----") (col (30), a);
  get file (datafile) list (num_sl_pp, num_size);
  do n = 1 to num_size;
    get file (datafile) list (num, size);
    put file (report) edit (num, size) (col (20), f (5, 2), col (30), f (5, 2));
  end;
  put file (report) skip (5);
end;

end;
put file (report) page;

end;
close file (report);
end hide_report;

```

```

/* .....
/*
/* Written in 1979 at Arthur D. Little, Inc.
/* Cambridge, Mass.
/*
/* .....
/* coded 10-15-79 by T. J. Martin
/* changed 2-20-80 for new PRA format
/* changed 3-15-80 to read surface impoundment data
/* changed 4-3-80 to handle TBS data correctly
/* changed 4-10-80 to read reg/nonreg & surface area
/* changed 9-30-80 for new ISS prices & stat_block
/* changed 10-15-80 to add other mining
/* changed 10-16-80 to fix breakevenstat_block
/* changed 11-21-80 to add breakeven on cost basis
/*

```

```
himpact: procedure;
```

```
/* program to analyze costs for high impact industries */
```

```

dcl (l, j, k, l, m, n, p, q, w, x, y, z, displw,
     num_sub_cat, sic, sicc, num_trg,
     tot_num_wastes, sl_present, argno init (1)) fixed bin,
     alien fixed bin (21),
     tr init (1), fr init (2), rr init (3), mt init (4), ad init (5), cp init (7), to init (6), nfr init (8),
     dcl breakeven (0:10) float dec static init (
       0e0, 5e3, 1e3, 6e3, 4e3, 1e3, 0e0, 0e0, 1e4, 1e3, 1e4);

dcl ec fixed bin (35),
     scale_factor float dec (6),
     bf float dec (6);

dcl (pl (10), loc (10)) fixed bin;
dcl ncosts (4, 10, 5, 4, 8) float dec (6),
     ntcosts (2, 5, 4, 8) float dec (6),
     nt (2, 5, 4, 8) float dec (6);

dcl best (5, 4, 8) float dec (6);
dcl (ann_ntl, ann_best) float dec (6);
     close_period float dec ext static,
     prodval float dec,
     vadd float dec,
     dp fixed bin,
     (num_plants, nplants, num_req_plants, num_nreq_plants, npararray (2)) float dec (6),
     tot_type_waste dec (6);

dcl price_multiplier float dec init (1e0);
dcl non_tech_array (5, 4, 8) float dec (6);
dcl costs (7) float dec (6);

dcl (aptr, tr_ptr) pointer;
/* costs from nontech */

```

```

dcl (seq_save, segment) char (64) var,
topstring (7) char (48) var ext static,
mp_name char (64) var,
sname char (64) var,
industry char (64) var,
date char (64) var,
filestr char (24) var init (""),
allstr char (20) init ("letxtorgainornonf"),
fn char (64) var,
arg char (alen) unaligned based (aptr), test_code char (1) var init ("");

dcl (iss, r78,
be_ok, be_sw, ths_sw, urq,
nostate, noslw, nowd, report, wvsw, cpq, price_mult, cbe, onslit, ofslit, slbit) bit (1) aligned init ("0'b");
dcl true bit (1) aligned static init ("1'b");
dcl ind_name char (84) var;

dcl names (6) char (50) var init
("Textiles: ", "Leather Tanning&Finishing: ",
"Non-Ferrous Metals: ",
"Organic Chemicals: ", "Inorganic Chemicals: ", "Other Mining:");

dcl 1 tech_reqts (num_trq) aligned based (tr_ptr),
2 num_plants float dec,
2 num_gIVEN fixed bin,
2 tr (6) fixed bin;

Xinclude nontech;

dcl num_codes fixed bin;

dcl 1 report_structs aligned ext static,
2 ind char (64) var,
2 num_models fixed bin,
2 state bit (1),
2 slsw bit (1),
2 nowd bit (1),
2 lss bit (1),
2 mod (68),
3 mp_name char (64) var,
3 scale_factor float dec (6),
3 num_sub_cat fixed bin,
3 sub (12),
4 sname char (64) var,
4 num_codes fixed bin,
4 code (7) fixed bin,
4 num_plants fixed bin,
4 costs (7) float dec (6),
4 tot_type_waste float dec (6),
4 displw fixed bin;

```

```

Xinclude cost;
dcl (lcost, bcost, ecost, clcost, om) float dec (6) init (0);

Xinclude rcra;

dcl endfile condition;

```



```

dc1 (datafile, datafile2) file stream input,
  sysprint file stream output;
dc1 (sum, float, substr, index, length) builtin;
dc1 waste_cost_ext entry (pointer, pointer) returns ((. . . .) float dec (5));
dc1 assign_cost_ext entry (pointer, pointer, fixed bin, fixed bin, fixed bin) returns ((. . . .) float dec (5));
dc1 annualize_npv_ext entry (float dec (5), float dec (5), float dec (5), float dec (5)) returns (float dec (5));
dc1 cu_sarg_ptr_ext entry (fixed bin, ptr, fixed bin (21), fixed bin (35));
%page;
  on endfile (datafile) begin;
    close file (datafile);
    if length (filestr) > 4 then filestr = substr (filestr, 5);
    else filestr = "";
    go to start;
  end;

argop:
  call cu_sarg_ptr (argno, aptr, alen, ec);
  if ec = 0 then do;
    if filestr = "" then do;
      put skip list ("An Industry must be specified in command.");
      return;
    end;
    else go to start;
  end;

  if arg = "-iss" then iss = true;
  else if arg = "-78" then r78 = true;
  else if arg = "-leather" | arg = "-leth" then filestr = filestr || "leth";
  else if arg = "-textile" | arg = "-text" then filestr = filestr || "text";
  else if arg = "-organic" | arg = "-orga" then filestr = filestr || "orga";
  else if arg = "-inorganic" | arg = "-inor" then filestr = filestr || "inor";
  else if arg = "-nonferrous" | arg = "-nonf" then filestr = filestr || "nonf";
  else if arg = "-mining" | arg = "-urom" then do;
    filestr = filestr || "urom";
    urog = true;
  end;

  if arg = "-cbe" then cbe = true;
  else if arg = "-all" & urog then filestr = filestr || "allstr";
  else if arg = "-all" then filestr = "allstr";
  nostate = true;
  toptstring (2) = "Without State Regulations";
  if arg = "-pm" | arg = "-price_mult" then do /* end nostate */
    price_mult = true;
    argno = argno + 1;
    call cu_sarg_ptr (argno, aptr, alen, ec);
    if ec = 0 then do;
      put skip list ("A Multiplier must be specified for -price_mult.");
      return;
    end;
    price_multiplier = float (arg);
  end;
  else if arg = "-nosl" then noslaw = true;
  else if arg = "-nowd" then nowd = true;
  else if arg = "-rep" then do;
    report = true;
    report_structs.num_models = 0;
    do i = 1 to 68;
      do j = 1 to 12;

```



```

do k = 1 to 7;
  report_struct$.mod (1).sub (j).costs (k) = 0e0;
  /* end k */
end;
report_struct$.mod (1).sub (j).tot_type_waste = 0e0;
/* end j */
/* end i */
/* end rep */

end;
if arg = "-cp" then do;
  cpg = true;
  argno = argno + 1;
  call cu_arg_ptr (argno, aptr, alen, ec);
  if ec = 0 then clos_period = 20;
  else if index (arg, "--") = 0 then do;
    argno = argno - 1;
    clos_period = 20;
  end;
  else clos_period = float (arg);
  /* end if */
  /* end cp */
end;
else if arg = "-wv" then wvsw = true;
else if arg = "-breakeven" | arg = "-be" then be_sw = true;
else do;
  put skip list ("Args are: -lss -cbe -78 -nostate -nosl -nowd <industry> -rep -cp -wv -be");
  put skip (2);
  return;
end;
argno = argno + 1;
go to arglp;

allocate arg_struct;
arg_struct.lss = lss;
arg_struct.wvsw = wvsw;
arg_struct.noslsw = noslsw;
arg_struct.nostate = nostate;
arg_struct.cbe = cbe;
arg_struct.price_multiplier = price_multiplier;
if -cpg then clos_period = 20;
allocate nontech_struct;
if (report & length (filestr) > 4) then do;
  put skip list ("Only one industry per call for reports.");
  put skip (2);
  return;
end;

if ~(lss|78) then lss = true;

do while (filestr ^= "");
  tbs_sw = true;
  fn = substr (filestr, 1, 4);
  if fn = "text" then do;
    be_ok = true;
    ind_name = names (1);
  end;
  else if fn = "leth" then do;
    be_ok = true;
    ind_name = names (2);
  end;
  else if fn = "nonf" then do;
    be_ok = true;
    ind_name = names (3);
  end;
  /* end textile do group */
  /* end leather do group */
  /* fn = "nonf" do */

```

```

end;
else
  if fn = "orga" then do;
    be_ok = "true";
    ind_name = names (4);
    tbs_sw = true;
  /* end nonferrous do */
end;
else
  if fn = "lnor" then do;
    be_ok = true;
    ind_name = names (5);
  /* organics do group */
  /* fn = "lnor" do */
end;
else
  if fn = "urom" then do;
    be_ok = "true";
    ind_name = names (5);
  /* end lnorganic */
end;
else
  do;
    put skip list ("An industry must be specified in command.");
    return;
  /* end other mining do */
  /* forgot to give an industry */
end;
/* end error do */

if report then do;
  report_struct.ind = ind_name;
  report_struct.state = "nostate";
  report_struct.nowd = nowd;
  report_struct.slsw = "noslsw";
  report_struct.lss = lss;
end;

arg_struct.tbs_sw = tbs_sw;
if be_ok & be_sw then arg_struct.be_sw = be_sw;
else arg_struct.be_sw = "0"b;
fn = fn||".hidata";
open file (datafile) title ("vfile_ "||fn);

get file (datafile) list (industry, date, sic);
/* main loop of program */

i = 1;
arg_struct.mod = i;
get file (datafile) list (mp_name, sicc, prodval, vadd, nplants, scale_factor, num_sub_cat);
segment = substr (mp_name, 1, index (mp_name, "-") - 1);
if report then do;
  report_struct.mod (i).mp_name = mp_name;
  report_struct.mod (i).scale_factor = scale_factor;
  report_struct.mod (i).num_sub_cat = num_sub_cat;
  report_struct.num_models = i;
end;

do while (true);
  num_hi_seg = num_hi_seg + 1;
  ind_struct (num_segments + num_hi_seg).ind_name =
    substr (ind_name, 1, index (ind_name, "-") + 1) || segment;
  ind_struct (num_segments + num_hi_seg).value_add (i) = 0e0;
  seg_save = segment;
  do while (segment = seg_save);
    ind_struct (num_segments + num_hi_seg).value_add (i) =
      ind_struct (num_segments + num_hi_seg).value_add (i) + vadd;
    do j = 1 to num_sub_cat;
      get file (datafile)
        list (scname, num_nreq_plants, num_req_plants, tot_num_wastes, num_tr0, sl_present);
      num_plants = num_nreq_plants + num_req_plants;
    end;
  end;
end;

```

```

If nostate then do;
  npararray (1) = 0; npararray (2) = 1;
end;
else do;
  npararray (1) = num_req_plants; npararray (2) = num_nreq_plants;
  do k = 1 to 5;
    do l = 1 to 8;
      arg_struct.wsr (k, l) = state_reqs (k, l) *
        num_req_plants/num_plants+num_nreq_plants/num_plants;
    end;
  end;
end;
allocate tech_reqts;
arg_struct.num_types = allocate_waste_ (tot_num_wastes); /* return number of disposal types */

nontech_struct.num_gen = 1;
nontech_struct.wsw = wsw;
nontech_struct.num_codes = 1;
nontech_struct.test (+).nwsa = 0;
do n = 1 to arg_struct.num (n).disp_code = 0 then
  nontech_struct.test (1).nwsa =
  nontech_struct.test (1).nwsa +
  arg_struct.num (n).num_wastes;
end;
nontech_struct.test (1).code = test_code;
call nontech_iss_ (nt_ptr, addr (nontech_array));
if sum (nontech_struct.test (+).nwsa) = 0 then
  nontech_array (4, ry, nfr) = nontech_array (4, ry, nfr) + nontech_struct.num_gen * 500

nontech_struct.num_gen = num_plants;
call decode_tr_ (num_trg, arg_struct.pct_tr (3), arg_struct.pct_tr (1), arg_struct.pct_tr (2));
/* 1 = closure, 2 = fence, 3 = runoff */
if sl_present > 0 then arg_struct.num_types = sl_info_ (arg_struct.num_types);
if report then do;
  report_struct.mod (1).sub (j).scname = scname;
  report_struct.mod (1).sub (j).num_plants = num_plants;
  report_struct.mod (1).sub (j).num_codes = arg_struct.num_types;
  report_struct.mod (1).sub (j).dispsw = dispsw;
end;
best (*, *, *) = 0e0;
arg_struct.sibit = "0"b;
if cbe then ncsts = waste_cost_ (as_ptr, nt_ptr);
else do;
  do n = 1 to arg_struct.num_types;
    dp = arg_struct.num (n).disp_code;
    if dp = 0 then do;
      if ~((up = 6) & dp = 7) & arg_struct.noslw then do;
        nontech_struct.tw = arg_struct.num (n).tot_waste;
        nontech_struct.disp_code = dp;
        ncsts = assign_cost_ (as_ptr, nt_ptr, arg_struct.num (n).whr, n);
        do x = 1 to 5;
          do y = 1 to 4;
            do z = 1 to 8;
              ncsts (arg_struct.num (n).whr, n, x, y, z) = ncsts (1,
x, y, z);
            end;
          end;
        end;
      end;
    end;
  end;
end;

```

```

end;
else do;
  arg_struct.num (n).whr = 3;
  do x = 1 to 5;
    do y = 1 to 4;
      do z = 1 to 8;
        ncsts (3, n, x, y, z) = 0e0;
      end;
    end;
  end;
end;

end;
end;

end;
if ~(arg_struct.sibit & noslw) then do;
  if cbe then do;
    ann_best = left;
    do k = 1 to arg_struct.num_types; /* check all onsite */
      do l = 1 to arg_struct.num_types;
        loc (l) = 3;
        end; /* end l */
        loc (k) = 1;
        nt (l, *, *, *) = 0e0;
        do x = 1 to 5;
          do y = 1 to 4;
            do z = 1 to 8;
              do l = 1 to arg_struct.num_types;
                nt (l, x, y, z) = nt (l, x, y, z) + ncsts (loc (l), l, x, y, z);
              end; /* end l */
            end; /* end z */
          end; /* end y */
        end; /* end x */
        ann_ntl = annualize_npv_ (sum (nt (l, *, fy, *)), sum (nt (l, *, cl, *)));
        if ann_ntl < ann_best then do;
          ann_best = ann_ntl;
          best (x, y, z) = nt (l, x, y, z);
          do x = 1 to arg_struct.num_types;
            pl (x) = loc (x);
          end; /* end x */
          /* end new best */
        end; /* end k */
      end; /* end arg_struct.sibit then do k = 1 to arg_struct.num_types; /* check all offsite if
no sl */

do l = 1 to arg_struct.num_types;
  loc (l) = 4;
  end;
  loc (k) = 2;
  nt (l, *, *, *) = 0e0;
  do x = 1 to 5;
    do y = 1 to 4;
      do z = 1 to 8;
        do l = 1 to arg_struct.num_types;
          nt (l, x, y, z) = nt (l, x, y, z) + ncsts (loc (l), l, x, y, z);
        end; /* end l */
      end; /* end z */
    end; /* end y */
  end; /* end x */
  ann_ntl = annualize_npv_ (sum (nt (l, *, fy, *)), sum (nt (l, *, ry, *)));

```

```

sum (nt (1, *, cap, *)), sum (nt (1, *, cl, *)))
if ann_nt1 < ann_best then do
  ann_best = ann_nt1
  best (x, y, z) = nt (1, x, y, z)
  do x = 1 to arg_struct.num_types
    pl (x) = loc (x)
  end
  /* end x */
  /* end ann_ */
end
/* end arg_struct.sibit */

do k = 1 to n-1
  /* check other cases */
  do l = k+1 to arg_struct.num_types
    do m = 1 to arg_struct.num_types
      loc (m) = arg_struct.num (m).whr
    end
    nt (*, *, *, *) = 0e0
    do x = 1 to 5
      do y = 1 to 4
        do z = 1 to 8
          do m = 1 to arg_struct.num_types
            loc (k) = 1
            loc (l) = 2
            nt (1, x, y, z) = nt (1, x, y, z) + ncsts (loc (m),
m, x, y, z)
            loc (l) = 1
            loc (k) = 2
            nt (2, x, y, z) = nt (2, x, y, z) + ncsts (loc (m),
m, x, y, z)
          end
        end
      end
    end
  end
end
end
end

do m = 1 to 2
  ann_nt1 = annualize_npv_ (sum (nt (m, *, 1, *)),
sum (nt (m, *, 2, *)), sum (nt (m, *, 3, *)), sum (nt (m, *, 4
, *)))
  if ann_nt1 < ann_best then do
    ann_best = ann_nt1
    best (x, y, z) = nt (m, x, y, z)
    do w = 1 to arg_struct.num_types
      pl (w) = arg_struct.num (w).whr
    end
    if m = 1 then do
      pl (k) = 1
      pl (l) = 2
    end
    else do
      pl (k) = 2
      pl (l) = 1
    end
  end
end
end
end

end
do k = 1 to arg_struct.num_types
  arg_struct.num (k).whr = pl (k)
end
end
/* end cbe */
else do

```



```

bf = left;
ann_best = left;
onsbit = -0'b;
offsbit = -0'b;
p = 0; q = 0; w = 0; m = 0;
do k = 1 to arg_struct.num_types;
  if arg_struct.num(k).disp_code = 0 then do;
    if arg_struct.num(k).whr = 3 then do;
      w = k;
      if arg_struct.num(k).size_code < 5 then m = k;
    end;
    onsbite = -1'b;
  end;
  else do;
    offsbit = -1'b;
    q = k;
    if w < wsw then do;
      if arg_struct.num(k).annual < bf then do;
        p = k;
        bf = arg_struct.num(k).annual;
      end;
    end;
  end;
end;
if m = 0 then m = w;
arg_struct.num(m).whr = 1;
end;
if offsbit then do;
  if p = 0 then p = q;
  arg_struct.num(p).whr = 2;
end;
do k = m, p;
  if (k = m & onsbite) | (k = p & offsbit) then do;
    nontech_struct.tw = arg_struct(k).tot_waste;
    nontech_struct.disp_code = arg_struct.num(k).disp_code;
    ntcosts = assign_cost_ (as_ptr, nt_ptr, arg_struct.num(k).whr, k);
    do x = 1 to 5;
      do y = 1 to 4;
        do z = 1 to 8;
          ncosts (arg_struct.num(k).whr, k, x, y, z) = ntcosts (2,
x, y, z);
        end;
      end;
    end;
  end;
end;
do x = 1 to 5;
  do y = 1 to 4;
    do z = 1 to 8;
      do m = 1 to arg_struct.num_types;
        best (x, y, z) = best (x, y, z) + ncosts (arg_struct.num(m).wh
r, m, x, y, z);
      end;
    end;
  end;
end;
do k = 1 to arg_struct.num_types;
  if arg_struct.num(k).whr = 1 | arg_struct.num(k).whr = 3 then do;

```

```

arg_struct.num(k).num_gen_ons = arg_struct.num(k).num_gen;
arg_struct.num(k).tot_waste_ons = arg_struct.num(k).tot_waste;

end;
else do;
arg_struct.num(k).num_gen_ons = 0;
arg_struct.num(k).tot_waste_ons = 0;

end;

end;
if report then do;
do k = 1 to arg_struct.num_types;
if arg_struct.num(k).whr = 1; arg_struct.num(k).whr = 3 then
report_struct.mod(i).sub(j).code(k) =
arg_struct.num(k).disp_code;
else report_struct.mod(i).sub(j).code(k) =
arg_struct.num(k).disp_code + 100;

end;
if _nostate then do;
lcost = lcost + arg_struct.wsr(4, i);
bcost = bcost + arg_struct.wsr(4, i);
ecost = ecost + arg_struct.wsr(4, i);
clcost = clcost + arg_struct.wsr(4, i);
om = om + arg_struct.wsr(4, i);
end;
/* end state do */

report_struct.mod(i).sub(j).costs(1) =
report_struct.mod(i).sub(j).costs(1) + lcost * 1e3;
report_struct.mod(i).sub(j).costs(2) =
report_struct.mod(i).sub(j).costs(2) + bcost * 1e3;
report_struct.mod(i).sub(j).costs(3) =
report_struct.mod(i).sub(j).costs(3) +
report_struct.mod(i).sub(j).costs(cap) + sum(non_tech_array(*, cap, *))

sum(best(*, cap, *));
report_struct.mod(i).sub(j).costs(4) =
report_struct.mod(i).sub(j).costs(4) + om * 1e3;

/* add non-technical costs to incremental technical costs */

do l = 1 to 7; /* all tax deductible */
report_struct.mod(i).sub(j).costs(4) =
report_struct.mod(i).sub(j).costs(4) + sum(non_tech_array(*, ry, l)

+ sum(best(*, ry, l)));
report_struct.mod(i).sub(j).costs(5) =
report_struct.mod(i).sub(j).costs(5) + sum(non_tech_array(*, fy, l)

+ sum(best(*, fy, l)));
report_struct.mod(i).sub(j).costs(6) =
report_struct.mod(i).sub(j).costs(6) + sum(non_tech_array(*, fy, l)

+ sum(best(*, fy, l)));
/* 1 loop */
report_struct.mod(i).sub(j).costs(5) =
report_struct.mod(i).sub(j).costs(5) +
sum(non_tech_array(*, cl, *)) + sum(best(*, cl, *));
report_struct.mod(i).sub(j).costs(7) =
report_struct.mod(i).sub(j).costs(7) + non_tech_array(4, fy, nfr)
+ best(4, fy, nfr);
report_struct.mod(i).sub(j).tot_type_waste =
report_struct.mod(i).sub(j).tot_type_waste
+ sum(arg_struct.num(*).tot_waste);
/* end if report do */

end;

call add_to_impact(i);

```

```

arg_struct.num(k).num_gen_ons = arg_struct.num(k).num_gen;
arg_struct.num(k).tot_waste_ons = arg_struct.num(k).tot_waste;
end;
else do;
arg_struct.num(k).num_gen_ons = 0;
arg_struct.num(k).tot_waste_ons = 0;
end;
end;
if report then do;
do k = 1 to arg_struct.num_types;
if arg_struct.num(k).whr = 1 | arg_struct.num(k).whr = 3 then
report_struct.mod(i).sub(j).code(k) =
arg_struct.num(k).disp_code;
else report_struct.mod(i).sub(j).code(k) =
arg_struct.num(k).disp_code + 100;
end;
if nostate then do;
lcost = lcost + arg_struct.wsr(4, i);
bcost = bcost + arg_struct.wsr(4, i);
ecost = ecost + arg_struct.wsr(4, i);
cicost = cicost + arg_struct.wsr(4, i);
om = om + arg_struct.wsr(4, i);
end;
/* end state do */
report_struct.mod(i).sub(j).costs(1) =
report_struct.mod(i).sub(j).costs(1) + lcost + lcost * le3;
report_struct.mod(i).sub(j).costs(2) =
report_struct.mod(i).sub(j).costs(2) + bcost + bcost * le3;
report_struct.mod(i).sub(j).costs(3) =
report_struct.mod(i).sub(j).costs(3) +
sum(best(*, cap, *));
report_struct.mod(i).sub(j).costs(4) =
report_struct.mod(i).sub(j).costs(4) + om * le3;
/* add non-technical costs to incremental technical costs */
do l = 1 to 7; /* all tax deductible */
report_struct.mod(i).sub(j).costs(4) =
report_struct.mod(i).sub(j).costs(4) + sum(non_tech_array(*, ry, l)
+ sum(best(*, ry, l)));
report_struct.mod(i).sub(j).costs(6) =
report_struct.mod(i).sub(j).costs(6) + sum(non_tech_array(*, fy, l)
+ sum(best(*, fy, l)));
/* i loop */
end;
report_struct.mod(i).sub(j).costs(5) =
report_struct.mod(i).sub(j).costs(5) +
sum(non_tech_array(*, cl, *)) + sum(best(*, cl, *));
report_struct.mod(i).sub(j).costs(7) =
report_struct.mod(i).sub(j).costs(7) + non_tech_array(4, fy, nfr)
+ best(4, fy, nfr);
report_struct.mod(i).sub(j).tot_type_waste =
report_struct.mod(i).sub(j).tot_type_waste
+ sum(arg_struct.num(7).tot_waste);
/* end if report do */
end;
call add_to_impact(i);

```

```

else do i = 1 to nwt;
  /* read from datafile */
  get file (datafile) list (wname, volume, dept, num_sl_pp, sa);
  if i = 1 then dispaw = dspt;
  k = 0;
  if num_codes ^= 0 then do;
    do j = 1 to num_codes;
      if arg_struct(dsp(j)).spec_code = dept then k = j;
    end;
  end;
  if k = 0 then do;
    num_codes = num_codes + 1;
    k = num_codes;
    arg_struct.dsp (num_codes).spec_code = dept;
    call translate_code_ (dspt, dc, onsite_sw);
    if ~(be_ok & be_sw) & ~cbe then do;
      if dc = 1 & onsite_sw = 0 then
        dc = 10;
    end;
    os (dc) = onsite_sw;
    arg_struct.dsp (k).disp_code = dc;
    gen_struct (dc).disp_code = dc;
    gen_struct (dc).gen = num_plants;
    if dc = 7 then do;
      gen_struct (dc).num_sl = 0;
      gen_struct (dc).surf_area = 0;
    end;
  end;
  else do;
    gen_struct (dc).num_sl = num_sl_pp;
    gen_struct (dc).surf_area = sa;
  end;
  /* end else do */
  if os (dc) = 1 then do;
    gen_struct (dc).gen_ons = num_plants;
  end;
end;
else
  dc = arg_struct.dsp (k).disp_code;
  gen_struct (dc).num_wastes = gen_struct (dc).num_wastes + 1;
  gen_struct (dc).tot_waste = gen_struct (dc).tot_waste + volume;
  if os (dc) = 1 then do;
    place (dc) = 3;
    gen_struct (dc).tot_waste_ons = gen_struct (dc).tot_waste_ons + volume;
    arg_struct.dsp (k).two = arg_struct.dsp (k).two + volume;
  end;
  else do;
    arg_struct.dsp (k).tw_offs = arg_struct.dsp (k).tw_offs + volume;
    place (dc) = 4;
  end;
  /* end i loop */
end;
arg_struct.num_codes = num_codes;
do i = 0 to num_disp_codes;
  if i = 6 | i = 7 then place (i) = 3;
  else do;
    if gen_struct (i).tot_waste < breakeven (i) then place (i) = 4;
    else place (i) = 3;
  end;
end;
nwt = 0;
do i = 0 to num_disp_codes;
  if gen_struct (i).gen > 0 then do;
    /* now count waste types */
    /* copy */

```



```

else do i = 1 to nwt;
  get file (datafile) list (uname, volume, dept, num_si_pp, sa);
  if i = 1 then dispaw = dspt;
  k = 0;
  if num_codes ^= 0 then do;
    do j = 1 to num_codes;
      if arg_struct(dsp(j)).spec_code = dept then k = j;
    end;
  end;
  if k = 0 then do;
    num_codes = num_codes + 1;
    k = num_codes;
    arg_struct.dsp(num_codes).spec_code = dspt;
    call translate_code_(dspt, dc, onsite_sw);
    if ~(be_ok & be_sw) & ~cbe then do;
      if dc = 1 & onsite_sw = 0 then
        dc = 10;
      end;
      os(dc) = onsite_sw;
      arg_struct.dsp(k).disp_code = dc;
      gen_struct(dc).disp_code = dc;
      gen_struct(dc).gen = num_plants;
      if dc = 7 then do;
        gen_struct(dc).num_si = 0;
        gen_struct(dc).surf_area = 0;
      end;
    end;
  else do;
    gen_struct(dc).num_si = num_si_pp;
    gen_struct(dc).surf_area = sa;
  end;
  if os(dc) = 1 then do;
    gen_struct(dc).gen_ons = num_plants;
  end;
end;
else
  dc = arg_struct.dsp(k).disp_code;
  gen_struct(dc).num_wastes = gen_struct(dc).num_wastes + 1;
  gen_struct(dc).tot_waste = gen_struct(dc).tot_waste + volume;
  if os(dc) = 1 then do;
    place(dc) = 3;
    gen_struct(dc).tot_waste_ons = gen_struct(dc).tot_waste_ons + volume;
    arg_struct.dsp(k).two = arg_struct.dsp(k).two + volume;
  end;
else do;
  arg_struct.dsp(k).tw_offs = arg_struct.dsp(k).tw_offs + volume;
  place(dc) = 4;
end;
end;
arg_struct.num_codes = num_codes;
do i = 0 to num_disp_codes;
  if i = 6 | i = 7 then place(i) = 3;
  else do;
    if gen_struct(i).tot_waste < breakeven(i) then place(i) = 4;
    else place(i) = 3;
  end;
end;
nwt = 0;
do i = 0 to num_disp_codes;
  if gen_struct(i).gen > 0 then do;
    /* now count waste types */
    /* copy */

```



```

end;
if n = 0 then do;
  nwt = nwt + 1;

  n = nwt;
  arg_struct.num (n).disp_code = 7;
  arg_struct.num (n).num_wastes = 1;
  arg_struct.num (n).num_gen = num_plants;
  arg_struct.num (n).tot_waste = 0;
  arg_struct.num (n).num_sl = nsl;
  arg_struct.num (n).surf_area = sa;
  arg_struct.num (n).num_gen_ons = num_plants;
  arg_struct.num (n).tot_waste_ons = 0;
  arg_struct.num (n).whr = 3;

end;
else do;
  arg_struct.num (n).num_wastes = arg_struct.num (n).num_wastes + 1;
  arg_struct.num (n).num_sl = nsl;
  arg_struct.num (n).surf_area = sa;

end;
return (nwt);
end sl_info;

Xpage;
translate_code_1 procedure (dispsw, disp_code, onsite_sw);
/* procedure to read PRA codes and translate into nontech codes */
dcl (dispsw, disp_code, onsite_sw) fixed bin;

onsite_sw = 0;
go to code (dispsw);

code (1):
code (28): code (29):
onsite_sw = 1;
code (2): code (30):
disp_code = 1;
return;

code (3): code (4):
onsite_sw = 1;
disp_code = 9;
return;

code (14):
code (13):
onsite_sw = 1;
disp_code = 8;
return;

code (7):
code (8):
onsite_sw = 1;
disp_code = 2;
return;

code (15):
code (16):
onsite_sw = 1;
disp_code = 3;
return;

code (9):
code (10):
onsite_sw = 1;
disp_code = 4;
return;

/* landfill */
/* dump */
/* secure landfill */
/* landfill */
/* incinerator w/ scrubber */
/* incinerator w/o scrubber */

```

```

code (5): code (18):
    onsite_sw = 1;
    disp_code = 6;
    return;
/* disposal surface impoundment */

code (19): code (20):
    onsite_sw = 1;
    disp_code = 7;
    return;
/* treatment surface impoundment */

code (21): code (25):
    onsite_sw = 1;
code (3): code (11): code (12): code (17):
code (22): code (23): code (24):
code (26): code (27):
    disp_code = 8;
    return;
/* recycling */

xpage;
end translate_code_;
add_to_impact: procedure (model);
/* procedure to manage external variable impact for various reports */

dcl (i, j, k, s) fixed bin,
    lg static fixed bin init (1),
    (model, c) fixed bin;

do s = 1 to 2;
    if nparray (s) = 0 then go to next;
    do i = 1 to 5;
        do j = 1 to 4;
            do k = 1 to 8;
                if j = 1&k<8 then impact$ (num_segments+num_hi_seg, s, lg, i, j, k) =
                    impact$ (num_segments+num_hi_seg, s, lg, i, j, k)+ (non_tech_array (i, j, k)+best
                    (i, j, k));
                scale_factor*nparray (s);
            else impact$ (num_segments+num_hi_seg, s, lg, i, j, k) =
                (non_tech_array (i, j, k)+best (i, j, k))*nparray (s)+
                impact$ (num_segments+num_hi_seg, s, lg, i, j, k);
            /* end k */
        end j;
    end i;
    /* end i */
/* unregulated, regulated */
/* optimize */
/* section */
/* init, ann, cap, clos */
/* activity */

scale_factor*nparray (s);
else impact$ (num_segments+num_hi_seg, s, lg, i, j, k) =
    (non_tech_array (i, j, k)+best (i, j, k))*nparray (s)+
    impact$ (num_segments+num_hi_seg, s, lg, i, j, k);
/* end k */
end j;
end i;
/* end s */

next;

do k = 1 to arg_struct.num_types;
    dp = arg_struct.num (k).disp_code;
    if dp = 1 | dp = 5 | dp = 10 then c = 3;
    else if dp = 8 then c = 4;
    else if dp = 9 then c = 2;
    else if dp = 0 then c = 1;
    else if dp = 2 then c = 5;
    else if dp = 3 then c = 6;
    else if dp = 4 then c = 7;
    else if dp = 6 | dp = 7 then c = 8;
    else c = 0;

```

```

if c /= 0 then do:
  stat_block$ (num_segments + num_hi_seg).size (i).gen_ons (c)
  + stat_block$ (num_segments+num_hi_seg).size (i).gen_ons (c)
  + arg_struct.num (k).num_gen_ons;
  stat_block$ (num_segments+num_hi_seg).size (i).gen_offs (c)
  + (arg_struct.num (k).num_gen - arg_struct.num (k).num_gen_ons);
  + stat_block$ (num_segments+num_hi_seg).size (i).waste_ons (c)
  + arg_struct.num (k).tot_waste_ons + num_plants + scale_factor;
  stat_block$ (num_segments + num_hi_seg).size (i).waste_offs (c)
  + (arg_struct.num (k).tot_waste - arg_struct.num (k).tot_waste_ons) + num_plants + scale_factor;
  + (arg_struct.num (k).tot_waste - arg_struct.num (k).tot_waste_ons) + num_plants + scale_factor;

  r;
  end;
  end;
  return;
  end add_to_impact;
  end himpact;

```

I-92

```

/* begin segment nontech.incl.plt */
/* last changed 10-16-80 add avg entry and avg_mode variable */
/* changed 4-7-81 add incr variable */
/* changed 4-30-81 add avgw2 variable, delete incr */
dc1 1 nontech_struct aligned based (nt_ptr),
    2 num_gen float dec (6),
    2 num_gen_ons float dec (6),
    2 num_gen_offs float dec (6),
    2 t1 float dec (6),
    2 t2 float dec (6),
    2 t3 float dec (6),
    2 num_wastes fixed bin,
    2 disp_code fixed bin,
    2 size_code fixed bin,
    2 num_codes fixed bin,
    2 test (5),
    3 code char (1) var,
    3 nusa fixed bin,
    2 size_var float dec,
    2 wsw bit (1),
    2 avg_mode bit (1),
    2 avgw2 float dec (6);

dc1 nontech_iss_ext entry (pointer, pointer),
    nontech_iss_onsite_ext entry (pointer, pointer),
    nontech_iss_offsite_ext entry (pointer, pointer),
    nontech_iss_incremental_ext entry (pointer, pointer),
    nontech_iss_avg_ext entry (pointer, pointer),
    nontech_ext entry (pointer, pointer);
dc1 nt_ptr pointer;
/* end segment nontech.incl.plt */

```



```

/* ..... */
/* Copyright (c) 1979 by Arthur D. Little, Inc. */
/* Cambridge, Mass. */
/* ..... */
/* ..... */
/* program to calculate nontechnical costs as an external procedure */
/* coded 12-6-79 by Thomas J. Martin */

nontech: proc;
    dcl tot_capacity fixed dec (10);
    dcl carry (5, 4, 8) float dec (12);
    dcl iss bit (1) aligned init ("b");
    dcl test_code char (1) var init ("D");
    %include nontech;

dcl nontech_output_ext entry (pointer, fixed dec (10, 0), fixed bin);

dcl sysprint file stream output external;
dcl datafile file stream input external;
%page;

open file (datafile) title ("%file_ nontech.data");
allocate nontech_struct;
num_gen_offs = 0e0;
tw_offs = 0e0;
num_codes = 1;
test (1).code = "b";
test (1).nwsa = 3;
life = 1;
wsws = "0*b";

loop:
get file (datafile) list (num_gen, num_gen_ons, disp_code, size_var = tot_capacity;
num_wastes = 3;
tw, two = float (tot_capacity);
if num_gen < 0 then do;
close file (datafile);
free nontech_struct;
put skip (5);
stop;
end;
if iss then
carry = nontech_iss_(nt_ptr);

call nontech_output_(addr (carry), tot_capacity, disp_code);
go to loop;
```

end nontech;

I-96


```

/* section 3001 calculations
/* ..... */

    if sum(nontech_struct.test(*).nusa) /= 0 then do;
        do i = 1 to nontech_struct.num_codes;
            carry (1, fy, ad) = carry (1, fy, ad)+num_gen*nontech_struct.test (1).nusa* (2+cler (gen)+2+tech (gen));
        end;
        if nontech_struct.test (1).code = "a" then carry (1, fy, mt) = num_gen*.9*nontech_struct.test (1).nusa+
            carry (1, fy, mt);
        else if nontech_struct.test (1).code = "b" then
            carry (1, fy, mt) = num_gen*.75*nontech_struct.test (1).nusa+800+carry (1, fy, mt);
        else if nontech_struct.test (1).code = "c" then
            carry (1, fy, mt) = num_gen*.50*nontech_struct.test (1).nusa+800+carry (1, fy, mt);
        else if nontech_struct.test (1).code = "d" then
            carry (1, fy, mt) = num_gen*.5e-1*nontech_struct.test (1).nusa+8e2+carry (1, fy, mt);
        else if nontech_struct.test (1).code = "e" then carry (1, fy, mt) = 0+carry (1, fy, mt);
        else carry (1, fy, mt) = num_gen*nontech_struct.test (1).nusa+4e2+ /* default */
            carry (1, fy, mt);
        carry (1, fy, ad) = carry (1, fy, ad)+num_gen*nontech_struct.test (1).nusa* (2+.2)+supv (gen);
        carry (1, fy, mt) = carry (1, fy, mt) + num_gen*nontech_struct.test (1).nusa*3014;
    end;
    carry (1, ry, ad) = carry (1, ry, ad)+ (cler (gen)+tech (gen))*num_gen;
/* ..... */
/* Section 3002 calculations
/* ..... */

    carry (2, fy, rr) = carry (2, fy, rr)+num_gen* (.5+cler (gen)+1+tech (gen)); /* ID Code */
/* ..... */
/* Section 3004 calculations
/* ..... */

end;

return;

xpage;

offsets: entry (nt_ptr, CPTR);

cptr = CPTR;
num_manifest = tw_offs / 2e1;
num_manifest = max (num_manifest, 4*num_gen_offs);
carry (*, *, *) = 0e0;
if disp_code = 0 then return;

/* ..... */
/* section 3001 calculations
/* ..... */

do i = 1 to nontech_struct.num_codes;
    carry (1, fy, ad) = carry (1, fy, ad) + num_gen_offs + nontech_struct.test (1).nusa*3* (cler (gen)+tech (gen));
end;

/* ..... */

```



```

/* section 3002 calculations
/* .....

carray (2, fy, ad) = carray (2, fy, ad)+num_gen_offs* (16*cier (gen)+19*tech (gen)); /* man, lbl systems */
carray (2, fy, ad) = carray (2, fy, ad)+num_gen_offs* (3*4)*supv (gen); /* general administration */
carray (2, ry, rr) = carray (2, ry, rr)+num_manifest* (.5*cier (gen)+.5*tech (gen))*0.05; /* reports */
carray (2, ry, rr) = carray (2, ry, rr)+num_manifest* (2*tech (gen)+1*cier (gen)); /* annual report */
carray (2, ry, ad) = carray (2, ry, ad)+num_manifest* (.333*cier (gen)); /* manifest */
carray (2, ry, ad) = carray (2, ry, ad)+num_gen_offs* (18*8)*supv (gen);
carray (2, ry, rr) = carray (2, ry, rr)+num_gen_offs* 110; /* ongoing review and filling */
carray (2, ry, rr) = carray (2, ry, rr)+num_gen_offs* (20*5*tech (gen)+5*supv (gen));

/* .....
/* Section 3003 calculations
/* .....

num_trans = 0;
carray (3, fy, rr) = carray (3, fy, rr)+ (2*tech (trans)+cier (trans))*num_trans;
carray (3, fy, ad) = carray (3, fy, ad)+ (4*tech (trans)+2*cier (trans))*num_trans; /* manifest system */
carray (3, fy, ad) = carray (3, fy, ad)+ (.25*tech (trans)+.5*cier (trans))*num_trans*2;
carray (3, ry, ad) = carray (3, ry, ad)+.75*num_manifest* (truck (trans)/12+truck (trans)/60);
carray (3, ry, ad) = carray (3, ry, ad)+.25*num_manifest* (truck (trans)/6+truck (trans)/60);
carray (3, ry, rr) = carray (3, ry, rr)+num_trans* (20+12*cier (trans));
carray (3, ry, ad) = carray (3, ry, ad)+num_trans* (8+4)*supv (trans);

/* .....
/* Section 3004 & 3005 calculations
/* .....

wlabel:
call waste_analysis;

if wsw then do;
  call get_size (disp_code, avgw2, tsize);
  tsize = min (4, tsize);
  annual_cost = tank_closure (tsize)*2;
  carray (4, cl, nfr) = carray (4, cl, nfr) + num_gen_offs*annual_cost;
  carray (5, fy, rr) = carray (5, fy, rr) + num_gen_offs* (1*tech (gen)+.5*cier (gen)); /* notify */
  carray (4, fy, cp) = carray (4, fy, cp) + num_gen_offs*8.6e1; /* security */
  carray (4, fy, ad) = carray (4, fy, ad) + num_gen_offs*1e2; /* inspection plan */
  carray (4, ry, ad) = carray (4, ry, ad) + num_gen_offs*1.5e3; /* inspection */
  carray (4, cap, mt) = carray (4, cap, mt) + num_gen_offs*0e0; /* air monitoring equip was 3500 */
  carray (4, fy, mt) = carray (4, fy, mt) + num_gen_offs*0e0; /* monitoring was 4000 */
  carray (4, cap, cp) = carray (4, cap, cp)+num_gen_offs*containment (tsize)*2; /* tank containment */
end;
return;

Xpage:
onsite:
  entry (nt_ptr, CPTR);

  cptr = CPTR;
  carray (., ., .) = 0e0;
  if disp_code = 0 then return;
  carray (4, fy, ad) = carray (4, fy, ad)+ (40*tech (gen)+30*cier (gen))*num_gen_ons; /* tr plan design */
  carray (4, fy, ad) = carray (4, fy, ad)+ (90*tech (gen)+32*cier (gen))*num_gen_ons; /* waste plan */
  carray (5, fy, rr) = carray (5, fy, rr)+ (1*tech (gen)+.5*cier (gen))*num_gen_ons; /* ID code */
  carray (5, fy, cp) = carray (5, fy, cp) + num_gen_ons* (30*supv (gen)+15*tech (gen)+10*cier (gen));
  carray (4, fy, tq) = carray (4, fy, tq) + 10175*num_gen_ons; /* training */
  carray (5, fy, rr) = carray (5, fy, rr)+num_gen_ons* (20*supv (gen)+40*tech (gen)+15*cier (gen)); /* part A */

```

```

carry (5, fy, ad) = carry (5, fy, ad)+num_gen_ons* (40*supv (gen)+20*tech (gen)+15*cler (gen));
carry (4, fy, ad) = carry (4, fy, ad)+num_gen_ons* (35*tech (gen)+15*cler (gen));
carry (4, fy, ad) = carry (4, fy, ad)+num_gen_ons* (15*supv (gen)+15*cler (gen));
if size_code < 5 then carry (4, fy, tg) = carry (4, fy, tg) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* records, operating log, quarterly rep, incidents, monitoring */
else carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* general overhead */
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* 10 emps for small */
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* 15 employees */
call waste_analysis;
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* recordkeeping for training */
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* periodic testing */
end;
else do;
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* annual report */
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* weekly inspections and recording */
if disp_code = 3 & disp_code = 4 then do;
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* disposal records */
end;
else do;
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* end do */
end;
end;
carry (4, fy, rr) = carry (4, fy, rr) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* end do */
do freq = 1 to 2;
carry (4, freq, mt) = carry (4, freq, mt) + num_gen_ons* (15*supv (gen)+15*cler (gen)); /* financial repts. & monitoring/testing */
end;
annual_cost = (closure_costs (disp_code, size_code, rv) + post_closure_costs (disp_code, size_code, rv));
annual_cost = annual_cost + closure_costs (disp_code, size_code, rv);
carry (4, cl, nfr) = carry (4, cl, nfr) + num_gen_ons* annual_cost;
carry (4, cap, cp) = num_gen_ons + 10000;
if disp_code = 3 | disp_code = 4 then carry (4, cap, mt) = carry (4, cap, mt) + num_gen_ons* 0;
else carry (4, cap, mt) = carry (4, cap, mt) + num_gen_ons* 23000; /* monitoring wells */
return;
xpage;
Incremental:
entry (nt_ptr, cptr);
/* calculate those costs incremental to a disposal method, rather than */
/* site specific */
cptr = CPTR;
num_manifest = tw_offs / 201;
carry (0, 0, 0) = 0;
if disp_code = 0 then return;
/* 3002 calculations */
carry (2, rv, rr) = carry (2, rv, rr) + num_manifest* (.5*cler (gen)+.5*tech (gen))+.05; /* reports */
carry (2, rv, rr) = carry (2, rv, rr) + num_gen_ons* (2*tech (gen)+1*cler (gen)); /* annual report */
carry (2, rv, ad) = carry (2, rv, ad) + num_manifest* (.333*cler (gen)); /* manifest */
/* 3003 calculations */
carry (3, rv, ad) = carry (3, rv, ad) + .75*num_manifest* (truck (trans)/12+truck (trans)+1.5e1/3.6e2);
carry (3, rv, ad) = carry (3, rv, ad) + .25*num_manifest* (truck (trans)/6+truck (trans)+1.5e1/3.6e2);
/* 3004 & 3005 calculations */
if num_gen_ons = 0 then do;
call waste_analysis;
return;
end;
carry (5, fy, rr) = carry (5, fy, rr) + num_gen_ons* (5*supv (gen)+10*tech (gen)+4*cler (gen)); /* part A */

```

```

carray (5, fy, ad) = carray (5, fy, ad)+num_gen_ons* (10*supv (gen)+5*tech (gen)+4*cler (gen));
carray (4, ry, mt) = carray (4, ry, mt)+num_gen_ons*ann_test (disp_code); /* periodic testing */
call waste_analysis;
carray (4, ry, rr) = carray (4, ry, rr)+num_gen_ons* (cler (gen)+tech (gen)); /* annual report */
carray (4, ry, ad) = carray (4, ry, ad) + num_gen_ons*5*tech (gen); /* weekly inspections and recording */
carray (4, ry, rr) = carray (4, ry, rr) + num_gen_ons* (12*tech (gen)+24*cler (gen)+20); /* disposal records */
if disp_code = 3 & disp_code = 4 then do;
    /* non-incinerator gw monitoring */
    carray (4, ry, rr) = carray (4, ry, rr)+num_gen_ons* (8*tech (gen)+3*cler (gen)+20);
end;
else do;
    carray (4, ry, rr) = carray (4, ry, rr)+num_gen_ons*2* (8*tech (gen)+2*supv (gen)+3*cler (gen)+20);
    /* end else do */
end;
do freq = 1 to 2;
    /* financial eqts. & monitoring/testing */
    carray (4, freq, mt) = carray (4, freq, mt)+num_gen_ons*mon_test_costs (disp_code, size_code, freq);
end;
/* end i loop */
annual_cost = (closure_costs (disp_code, size_code, ry)+post_closure_costs (disp_code, size_code, ry));
annual_cost = annual_cost + closure (disp_code, size_var);
carray (4, cl, nfr) = carray (4, cl, nfr)+num_gen_ons*annual_cost;
carray (4, cap, cp) = num_gen_ons * 7500;
if disp_code = 3|disp_code = 4 then carray (4, cap, mt) = carray (4, cap, mt)+num_gen_ons*0;
else carray (4, cap, mt) = carray (4, cap, mt)+num_gen_ons*23000; /* monitoring wells */
return;

%page;
avg:
entry (nt_ptr, CPTR);

/* special ISS costs for small waste generators */

cptr = CPTR;
carray (0, 0, 0) = 0e0;
if disp_code = 0 then return;

if num_gen_offs > 0 then do;
    call waste_analysis;
    if avg_mode then do;
        if wsw then do;
            call get_size (disp_code, avgw2, tszsize);
            tszsize = min (4, tszsize);
            annual_cost = tank_closure (tszsize)+2*te2;
            carray (4, cl, nfr) = carray (4, cl, nfr) + num_gen_offs*annual_cost;
            carray (5, fy, rr) = carray (5, fy, rr) + num_gen_offs* (1*tech (gen)+5*cler (gen)); /* notify */
        end;
        carray (4, fy, cp) = carray (4, fy, cp) + num_gen_offs*8.6e1; /* security */
        carray (4, fy, ad) = carray (4, fy, ad) + num_gen_offs*te2; /* inspection plan */
        carray (4, ry, ad) = carray (4, ry, ad) + num_gen_offs*1.5e3; /* inspection */
        carray (4, cap, mt) = carray (4, cap, mt) + num_gen_offs*0e0; /* air monitoring equip was 3500 */
        carray (4, fy, mt) = carray (4, fy, mt) + num_gen_offs*0e0; /* monitoring was 4000 */
        carray (4, cap, cp) = carray (4, cap, cp)+num_gen_offs*containment (tszsize)+2; /* tank containment
    */
end;
end;

end;

return;

%page;
waste_analysis: procedure;

/* subroutine to calculate waste analysis costs for offsite disposers */
avg_waste = tw_offs / num_wastes;
if avg_waste <= 80 then
    /* waste analysis - ants in tonnes */

```

```

carray (4, ry, tr) = carray (4, ry, tr) + num_gen_offs*num_wastes*6e2*pfactor;
else if av0_waste < 160 then
  carray (4, ry, tr) = carray (4, ry, tr) + num_gen_offs*num_wastes*av0_waste* 8e2/1.6e2*pfactor
  fav0_waste*5e1/2e1*num_wastes*num_gen_offs*pfactor;
else carray (4, ry, tr) = carray (4, ry, tr) + num_gen_offs*num_wastes* 5e1/2e1*av0_waste*pfactor +
  8e2*num_wastes*num_gen_offs*pfactor;
end waste_analysis;

end nontech_iss_;

```



```

/* .....
* .....
* Copyright (c) 1978 by Arthur D. Little, Inc.
* .....
* .....
*/

```

```

nontech_output_: procedure (CARRY_PTR, TOI_WASTE, TYPE_CODE);

```

```

  dcl carry (5, 4, 8) based (ca_ptr) float dec (12);
  dcl TOI_WASTE fixed dec (10, 0), TYPE_CODE fixed bin;
  dcl prev_inv fixed bin static init (0);
  dcl (js, freq, smt, i, j, nsec, num_segments) fixed bin;

```

```
/* used for header */
```

```

  dcl col_name (7) char (8) static init ("fin resp", "rec/rep", "mon/test", "admin", "training", "contin", "non-ded");
  dcl type_name (7) char (32) varying init ("Landfill", "Landfarm", "Incinerator (Chl)", "Incinerator (Non)",
    "Monofill", "Surf. Imp. (Land Cl)", "Surf. Imp. (Sludge Rem.)");

```

```

  dcl (ltrim, char, sum) builtin;

```

```

  dcl (CARRY_PTR, ca_ptr) pointer;
  dcl (sysprint output, report output) file stream;

```

```
/* ..... */
```

```
/* output & report section */
```

```

  ca_ptr = CARRY_PTR;

```

```

  if prev_inv = 0 then do;

```

```

    put edit ("Non-Technical Costs Per Facility") (col (40), a);

```

```

    put skip;

```

```

    put edit ("-----") (col (40), a);

```

```

    put skip (3);

```

```

    put edit ((col_name (1) do 1 = 1 to 7)) (x (10), (7) (x (5), a (10)));

```

```

    put skip (2);

```

```

    prev_inv = 1;

```

```

  end;

```

```

  put edit (type_name (TYPE_CODE), TOI_WASTE (col (10), a, x (2), p'zz.zzz.zzz'));

```

```

  put skip;

```

```

  ));

```

```

  put skip edit ("Capital:", (sum (carry (*, 3, 1)) do 1 = 2 to 8)) (x (1), a (8), x (9), (7) (x (5), p'$$.$$$.$$$

```

```

  ));

```

```

  put skip edit ("Initial:", (sum (carry (*, 1, 1)) do 1 = 2 to 8)) (x (1), a (8), x (9), (7) (x (5), p'$$.$$$.$$$

```

```

  ));

```

```

  put skip edit ("Closure:", (sum (carry (*, 4, 1)) do 1 = 2 to 8)) (x (1), a (8), x (9), (7) (x (5), p'$$.$$$.$$$

```

```

  ));

```

```

  put skip edit ("Annual:", (sum (carry (*, 2, 1)) do 1 = 2 to 8)) (x (2), a (7), x (9), (7) (x (5), p'$$.$$$.$$$

```



```

put skip (2);

/* put skip(2) edit((col_name(i) do i -1 to 8)) (x(16).(8)(a(11),x(3)))
do i = 1 to 5;
put skip edit(("300"||trim(char(i))||"-".(foo(1,j) do j=1 to 7) do i=1 to 2)) ((2)(a(5),x(2),(8)(x(2),f(7,3)),x(5))))
end; */

end nontech_output_i

```

npv.plt

11/01/80 1336.6r w 11/01/80 1330.0

100332

npv: procedure;

Xinclude rcra;

```

dc1 clos_period float dec ext static;
dc1 disc static float dec init (.14);
dc1 clos_pat float dec;
dc1 rt float dec init (2.4e-1) ;

```

```

dc1 1 rep_struc (num_segments + hl_incr) aligned based (rep_ptr),
    2 name char (60) var,
    2 tva float dec,
    2 tcap float dec,
    2 tlcost float dec,
    2 npv float dec,
    2 frpt float dec,
    2 ltmt float dec,
    2 tfye float dec,
    2 taeo float dec,
    2 ttac float dec;

```

```

dc1 list (counter) based (list_ptr) fixed bin;
dc1 (list_ptr, rep_ptr) pointer;
dc1 outputs (4, 5, 124, 8) ext static fixed dec (9, 3),
    (cap, clcost, tye, aoe, od, tac, npv, frpt, ltmt, va, num_gen, alloc) float dec,
    (tcap, tfye, taeo, tod, ttac, tva) float dec,
    (gtcap, gttye, gtaeo, gtod, gtac, gtva) float dec;
dc1 (freq, sat, mod, l, j, nsec, alien, argno init (1), num_models static init (4)) fixed bin,
    hl_incr fixed bin init (0),
    counter fixed bin init (0),
    ec fixed bin (35),
    (month, dom, year, hour, minute, second, dow) fixed bin,
    (bits, microsecond) fixed bin (71),
    code fixed bin (35),
    sun fixed bin init (1);

```

```

dc1 str1 char (64) var init ("Incremental Cost of RCRA Interim Status Standards"),
    arg char (alen) unaligned based (aptr),
    option char (15) var,
    opstr char (100) var init ("cap clos fyexp annex npv first_imp lt_imp"),
    str2, old_ind, industry, ind_save) char (60) var,
    filler char (3) var init (" "),
    zone char (3) init (" "),
    shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT"),
    mo (12) char (10) var init ("January", "February", "March", "April", "May", "June", "July", "August", "September",
    "October", "November", "December"),
    days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");

```

```

dc1 (more_sat, repeat) bit (1) aligned init ("1b");
dc1 (h1sw, industry_sw, order, up) bit (1) aligned init ("0b");
dc1 (aptr, optr) pointer;

```

```

dc1 (substr, index, sum, length, char, clock, ltrim) builtin;

```

```

endpage condition,
(sysprint output, report output) file stream,
cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35));
dc1 decode_clock_value_date_time entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
fixed bin, char (3), fixed bin (35));
dc1 header_ext entry (char (64) var, file);
xpage;

on endpage (report) begin;
  put page file (report);
  put file (report) edit (mo (month), dom, " ", year, hour, ":",
  filler, trim (char (minute)), shift (sun), days (dow))
  col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
  x (1), f (2), a (1), a, a, x (1), a, x (1), a (3);
  put file (report) skip;
  call header_ ("NET PRESENT VALUE REPORT", report);
  put file (report) skip (3);
  if ~industry_sw then put file (report) edit ("Industry:") (col (5), a);
  put file (report) edit ("1978 Value") (col (34), a)
  ("") (col (47), a)
  ("First Year") (col (57), a)
  ("Closure") (col (70), a)
  ("Annual") (col (83), a)
  ("Net Present") (col (93), a)
  ("First Year") (col (108), a)
  ("Long-term") (col (123), a);
  put file (report) skip;
  if industry_sw then put file (report) edit ("Industry") (col (5), a);
  else put file (report) edit ("Segment") (col (5), a);
  put file (report) edit ("Added") (col (36), a)
  ("Capital") (col (47), a)
  ("Expenses") (col (58), a)
  ("Costs") (col (71), a)
  ("Expenses") (col (82), a)
  ("Value") (col (96), a)
  ("Impact") (col (110), a)
  ("Impact") (col (124), a);
  put file (report) skip;
  if industry_sw then put file (report) edit ("") (col (5), a);
  else put file (report) edit ("") (col (5), a);
  put file (report) edit ("") (col (34), a)
  ("") (col (47), a)
  ("") (col (57), a)
  ("") (col (70), a)
  ("") (col (82), a)
  ("") (col (93), a)
  ("") (col (108), a)
  ("") (col (123), a);
  put file (report) skip (2);
end;
/* begin block */

```

```

arglp:
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to proceed;
else if arg = "--impact" | arg = "--hi" then hlsu = "1'b";
else if arg = "--industry" | arg = "--ind" then industry_sw = "1'b";
else if arg = "--sort" then do;
  order = "1'b";

```

```

argno = argno + 1;
call cu_sarg_ptr (argno, sptr, alen, ec);
if ec = 0 then option = "pctva";
else if index (arg, "-") = 0 then do;
    argno = argno - 1;
    option = "pctva";
end;
else if index (opstr, arg) = 0 then do;
    put skip edit ("Sort options are:", opstr) (a, x (2), a);
    put skip (2);
    return;
end;
else option = arg;
end;
else if arg = "-asc" | arg = "-desc" then do;
    if "order" then do;
        put skip list ("If sort desired, list as argument.");
        put skip list ("Args -desc and -asc invalid without sort arg.");
        return;
    end;
    else if arg = "-asc" then up = "b";
    end;
    else do;
        put skip list ("Args are: -himpact -industry");
        put skip (2);
        return;
    end;
    argno = argno + 1;
    go to arglp;
/* end error do */

proceed;

if hrow then hl_incr = num_hl_seg;
allocate rep_struct;
open file (report) title ("vfile_ npv.report") print linesize (135);

bits = clock ();
call decode_clock_value_sdate_time
(bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code);
if hour = 12 then do;
    if minute = 0 then sun = 3;
    else sun = 2;
end;
else if hour = 0 then do;
    hour = 12;
    if minute = 0 then sun = 4;
end;
else if hour > 12 then do;
    hour = hour - 12;
    sun = 2;
end;
if minute < 10 then filler = "0";

outputs (" ", " ", " ") = 0;
do smt = 1 to num_segments + hl_incr;
    rep_struct (smt).tva = 0e0;
    rep_struct (smt).tcap = 0e0;
    rep_struct (smt).tclcost = 0e0;
    rep_struct (smt).npv = 0e0;
    rep_struct (smt).fmpt = 0e0;

```



```

rep_struc (smt).ltimept = 0e0;
rep_struc (smt).tfye = 0e0;
rep_struc (smt).taoe = 0e0;
rep_struc (smt).ttac = 0e0;
do nsec = 1 to 5;
  do freq = 1 to 4;
    do i = 1 to 8;
      do mod = 1 to num_models;
        if smt <= num_segments then
          outputs (freq, nsec, smt, i) =
            state_regs (nsec, i) * carry$ (smt, 2, mod, nsec, freq, i) +
            state_regs (freq, nsec, smt, i) * carry$ (smt, 1, mod, nsec, freq, i) / 1e6;
        else
          outputs (freq, nsec, smt, i) =
            state_regs (freq, nsec, smt, i) + (impact$ (smt, 2, mod, nsec, freq, i) +
            state_regs (freq, nsec, smt, i) * impact$ (smt, 1, mod, nsec, freq, i) / 1e6);
        /* end mod */
      end;
    end;
  end;
end; end; end;

signal endpage (report);
smt = 1;
str2 = ind_struc$ (smt).ind_name;
industry = substr (str2, 1, index (str2, ":")-1);
ind_save = industry;
do while (more_smt);
  counter = counter + 1;
  if industry_sw then rep_struc (counter).name = industry;
  else rep_struc (counter).name = str2;
  do while (industry = ind_save & more_smt & repeat);
    va = sum (ind_struc$ (smt).value_add (*));
    cap = sum (outputs (3, *, smt, *));
    fye = sum (outputs (1, *, smt, *));
    clocost = sum (outputs (4, *, smt, *));
    aoe = sum (outputs (2, *, smt, *));
    rep_struc (counter).tva = rep_struc (counter).tva + va;
    rep_struc (counter).tcap = rep_struc (counter).tcap + cap;
    rep_struc (counter).tclcost = rep_struc (counter).tclcost + clocost;
    rep_struc (counter).tfye = rep_struc (counter).tfye + fye;
    rep_struc (counter).taoe = rep_struc (counter).taoe + aoe;
    smt = smt + 1;
    old_ind = str2;
    if smt <= num_segments + hl_incr then do;
      str2 = ind_struc$ (smt).ind_name;
      industry = substr (str2, 1, index (str2, ":")-1);
    end;
    else more_smt = "0"b;
    repeat = industry_sw;
  end;
  rep_struc (counter).npv = rep_struc (counter).tfye + rep_struc (counter).npv + rep_struc (counter).tclcost +
  /3.716e-1);
  do i = 0 to 9;
    rep_struc (counter).npv = rep_struc (counter).npv + rep_struc (counter).taoe / (1+d1ec)*i;
  end;
  if rep_struc (counter).tva = 0 then do;
    clos_pmt = rep_struc (counter).tclcost / 3.716e-1;
    clos_pmt = clos_pmt + rtv (1+rt)*clos_period;
    clos_pmt = (clos_pmt / ((1 + rt)*clos_period - 1));
    rep_struc (counter).fmpmt = (rep_struc (counter).tcap + clos_pmt +
    rep_struc (counter).ltimept + rep_struc (counter).taoe) / rep_struc (counter).tval;
    rep_struc (counter).ltimept = rep_struc (counter).taoe / rep_struc (counter).tval;
  end;

```



```

else do;
  rep_struc (counter).fmpst = 0;
  rep_struc (counter).ltmpt = 0;
end;
ind_save = industry;
repeat = "1b";
end;
allocate list;
if order then call sort (option, up);
else do i = 1 to counter;
  list (i) = i;
end;
end;

do i = 1 to counter;
  if (industry sw) then put file (report) skip edit
    (rep_struc (list (i)).name) (col (1), a (28));
  else put file (report) skip edit (substr
    (rep_struc (list (i)).name, 1, index (rep_struc (list (i)).name, "1b")))
    (a)
    (substr (rep_struc (list (i)).name, index (rep_struc (list (i)).name, "1b") +2))
    (skip, a (35));
  put file (report) edit
    (rep_struc (list (i)).tva) (col (33), p"zzz.z29v.99")
    (rep_struc (list (i)).tcap) (col (43), p"zzz.z29v.99")
    (rep_struc (list (i)).tfty) (col (54), p"zzz.z29v.99")
    (rep_struc (list (i)).tclcost) (col (66), p"zzz.z29v.99")
    (rep_struc (list (i)).taoe) (col (79), p"zzz.z29v.99")
    (rep_struc (list (i)).npv) (col (93), p"zzz.z29v.99")
    ((rep_struc (list (i)).fmpst)*1e2) (col (111), p"z9v.99")
    ((rep_struc (list (i)).ltmpt)*1e2) (col (125), p"z9v.99");
end;

/* print summary page for all industries */

put file (report) skip (2);
npv = sum (rep_struc (*).tfty) + sum (rep_struc (*).tcap) +
  (sum (rep_struc (*).tclcost) / 2.716e-1);
do i = 0 to 9;
  npv = npv + sum (rep_struc (*).taoe) / (1+d1sc)*.1;
end;
if sum (rep_struc (*).tva) ^= 0 then do;
  clos_pmt = sum (rep_struc (*).tclcost) / 2.716e-1;
  clos_pmt = clos_pmt + rt * ((1+rt)*clos_period);
  clos_pmt = clos_pmt / ((1+rt)*clos_period - 1);
  fmpst = (sum (rep_struc (*).tcap) + clos_pmt +
    sum (rep_struc (*).tfty) + sum (rep_struc (*).taoe)) / sum (rep_struc (*).tva);
  ltmpt = sum (rep_struc (*).taoe)/sum (rep_struc (*).tva);
end;
do;
  fmpst = 0;
  ltmpt = 0;
end;

put file (report) skip edit
  ("Total") (col (1), a (28))
  (sum (rep_struc (*).tva)) (col (32), p"zzz.z29v.99")
  (sum (rep_struc (*).tcap)) (col (43), p"zzz.z29v.99")
  (sum (rep_struc (*).tfty)) (col (54), p"zzz.z29v.99")
  (sum (rep_struc (*).tclcost)) (col (66), p"zzz.z29v.99")

```

```

(sum (rep_struct (*).tase)) (col (79), p'zzz.zzv.99")
(npv) (col (93), p'zzz.zzv.99")
((fmp1).ie2) (col (111), p'zv.99")
((ltmpt).ie2) (col (125), p'zv.99");
put file (report) skip;

close file (report);

free rep_struct;
free list;

```

Xpage;

sort; procedure (option, up);

dc1 option char (15) vari;

dc1 up bit (1) aligned;

dc1 item (counter) float dec;

dc1 bound float dec;

dc1 (i, j, k) fixed bin;

if option = "cap" then do i = 1 to counter;

item (i) = rep_struct (i).tcap;

end;

else if option = "clos" then do i = 1 to counter;

item (i) = rep_struct (i).tclcost;

end;

else if option = "fyexp" then do i = 1 to counter;

item (i) = rep_struct (i).tfys;

end;

else if option = "annexp" then do i = 1 to counter;

item (i) = rep_struct (i).tase;

end;

else if option = "npv" then do i = 1 to counter;

item (i) = rep_struct (i).npv;

end;

else if option = "first_imp" then do i = 1 to counter;

item (i) = rep_struct (i).fmp1;

end;

else if option = "lt_imp" then do i = 1 to counter;

item (i) = rep_struct (i).ltmpt;

end;

bound = item (i);

k = 1;

do i = 1 to counter;

do j = 1 to counter;

if (up & item (j) < bound) | (~up & item (j) > bound) then do;

bound = item (j);

k = j;

end;

end;

list (i) = k;

if up then do;

item (k) = 1000000000;

bound = 1000000000;

end;

else do;

item (k) = -1;

```
bound = -1;  
end;  
end;  
return;  
end sort;  
end npv;  
end
```

rcra.incl.plt

08/21/81 1335.0rew 08/14/81 1604.7

10287

```

/* begin segment rcra.incl.plt */
/* last modified 10-16-80 to distinguish incinerators */
dcl num_segments fixed bin static init (67);
dcl num_disp_codes fixed bin static init (10);
dcl 1 ind_structs (140) aligned external static,
  2 ind_name char (60) varying,
  2 sic fixed bin,
  2 value_add (4) float dec;
/* passes information to rcra_output_ */

dcl 1 stat_blocks (140) aligned ext static,
  2 size (4),
  3 gen_ons (8) float dec(6),
  3 gen_offs (8) float dec(6),
  3 waste_ons (8) float dec(6),
  3 waste_offs (8) float dec(6);
/* passes information to rcra_stat */

dcl arrays (67, 2, 4, 5, 4, 8) ext static float dec (6); /* seq. state, size, sect, (init, ann, cap, clos), activity */
dcl impact$ (68:140, 2, 4, 5, 4, 8) ext static float dec (6);
dcl sicost$ (67, 2, 2, 4, 4) ext static float dec (6); /* seq.('sl,sl).state,size,(l,e,o,c) */
dcl num_hf_seg ext static fixed bin;

dcl state_regs (5, 8) float dec init (
  .35, 1.0, (5).35, 1.0,
  .44, 1.0, (5).44, 1.0,
  .3, 1.0, (5).3, 1.0,
  .75, 1.0, (5).75, 1.0,
  .85, 1.0, (5).85, 1.0);

/* end segment rcra.incl.plt */

```

```

/* .....
/*
/* Written in 1980 at:  Arthur D. Little, Inc.
/*                      Cambridge, Mass. 02140
/* .....
/* .....

rcra_cost: procedure;

dcl nt2 (5, 4, 8) float dec (6);
dcl nta (5, 4, 8) float dec (6);
dcl ncosts (8, 5, 4, 8) float dec (6);
dcl maxsize (0:num_disp_codes) float dec (6);
dcl tank_closure (0:4) static float dec init (0e0, 1e3, 1.6e3, 2.2e3, 4.4e3);
dcl annual (10, bf) float dec;
dcl non_tech_array (5, 4, 8) float dec (8);
dcl costs (2, 5, 4, 8) float dec (8);
dcl (t1, t2) float dec;
dcl las_offs (0:num_disp_codes) float dec init (0e0,
7.45,
0.79,
1.832e1,
5.74,
7.45,
0.0,
9.33,
6.74,
9.33);
dcl codes (5) char (1) var static aligned init ("a", "b", "c", "d", "e");
dcl code_grp (5) fixed bin init (50);
dcl tsize fixed bin;
dcl (aptr, ptruc_ptr, wstruc_ptr) pointer, arg char (alen) unaligned based (aptr),
cutoff_fixed bin;
dcl topatring (7) char (48) var ext static;
dcl (freq, smt, l, j, n, nsec, nstat, argno init (1), act,
alen, num_smt, nmodels static init (4), num_disp_types,
sc, ss, (10), k, m, nmgr, nvs, dp, r, p, q, w, t, x, y, z, whr (10)) fixed bin;
dcl (tr init (1), fr init (2), rr init (3), mt init (4), ad init (5), cp init (7), tg init (8), nfr init (8),
fy init (1), ry init (2), cap init (3), ci init (4)) static fixed bin;
dcl size_var (10) float dec;
dcl (gon, waste) float dec (8);
dcl clos_period float dec ext static init (20),
price_multiplier float dec init (1e0),
(tprod, temp, ts_plus_both, di_plus_both, sa_per_ts) float dec;
dcl breakeven (0:num_disp_codes) float dec init (0e0, 5e3, 1e3, 8e3, 4e3, 1e3, 0e0, 0e0, 1e4, 1e3, 1e4);
dcl (all_ons, all_offs, iss, r78, nowd, noslaw, cutoff, price_mult, wsw, swgsw, offabit, onebit) bit (1) aligned init ("0",
onsite (2, 14, 7) bit (1) aligned,
true bit (1) static aligned init ("1b");

```



```

Xinclude nontech;

dcl 1 test (2, 7) aligned,
  2 number fixed bin,
  2 atcode (5),
  3 cd char (1) var,
  3 nusa fixed bin;

/* for use with allocate_waste */

dcl 1 prac_struct (num_disp_codes+1) aligned,
  2 disp fixed bin,
  2 num_wastes fixed bin,
  2 num_gen float dec (6),
  2 num_gen_ons float dec (6),
  2 num_gen_offs float dec (6),
  2 tot_waste float dec (6),
  2 tot_waste_ons float dec (6),
  2 tot_waste_offs float dec (6),
  2 num_si float dec (6),
  2 surf_area float dec (6);

Xinclude rcra;

dcl 1 wstruct (num_segments) aligned based (wstruct_ptr),
  2 num_tech fixed bin,
  2 si (2),
  3 num_groups fixed bin,
  3 group (7),
  4 pct float dec,
  4 orig_pct float dec,
  4 tech (14) fixed bin;

dcl 1 wload (2, nmodels, 14) float dec (6),
  sum_wload float dec (6);

/* structure for model plants */

dcl 1 pstruct (70) aligned based (pstruct_ptr),
  2 industry char (60) var,
  2 value_add float dec,
  2 pdata (2, 4),
  3 plants float dec (6),
  3 employment float dec (6),
  3 prod float dec (6),
  3 vol_waste float dec (6),
  3 level_one float dec (6),
  2 cutoff (3),
  3 waste float dec (6),
  3 employment float dec (6),
  3 plants float dec (6),
  2 wastes fixed bin,
  2 waste (14),
  3 ename char (60) var,
  3 volume float dec (6),
  3 cprac (3) fixed bin,
  3 cpct (3) float dec,
  3 new_prac (2) fixed bin,
  3 new_pct (2) float dec,
  3 atcode char (1) varying,
  3 pct_ons float dec,
  2 sla (2, 4),
  3 no_imp float dec (6),

```

```

3 disp_imp float dec (6),
3 ts_imp float dec (6),
3 both_imp float dec (6),
3 num_disp_pp float dec (6),
3 disp_sa float dec (6),
3 num_ts_pp float dec (6),
3 ts_sa float dec (6);

dcl (index, trunc, float, max, sum) builtin,
(sysprint output, datafile input, datafile2 input) file stream,
endfile condition;

dcl runoff_ext entry (fixed bin, float dec) returns (float dec),
fence_ext entry (fixed bin, float dec) returns (float dec),
closure_ext entry (fixed bin, float dec) returns (float dec),
get_size_ext entry (fixed bin, float dec (6), fixed bin),
cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35));

xpage;
do i = 1 to 7;
    topstring (1) = "";
end;
topstring (5) = "Including Surface Impoundments"; /* end 1 */

arglp:
call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to start; /* got all arguments */

if arg = "-iss" then do;
    iss = true;
end;
topstring (1) = "Interim Status Standards";

else if arg = "-78" then r78 = true;
else if arg = "-offs" then all_offs = true;
else if arg = "-ons" then all_ons = true;
else if arg = "-nosl" then do;
    noslaw = true;
end;
topstring (5) = "";

else if arg = "-vw" then do;
    wsw = true;
end;
topstring (4) = "Including Treatment Costs";

else if arg = "-nowd" then nowd = true;
else if arg = "-sug" then do;
    sugsw = true;
end;
topstring (4) = topstring (4) || " & SWG's"; /* option b2 */

else if arg = "-cutoff" | arg = "-cu" then do; /* end do */
    cutoff = true; /* cutoff */
    argno = argno + 1;
    call cu_sarg_ptr (argno, aptr, alen, ec);
    if ec = 0 then do; /* error */
        put skip list ("Specify cutoff amount: 0/100/500/1000");
        put skip (2);
        return;
    end;
    if arg = "0" | arg = "100" | arg = "500" | arg = "1000" /* end cutoff error */
    then cutoff_amt = arg;
    else go to cutoff_err;
end;

```

```

end;
else if arg = "-cp" then do;
    argno = argno + 1;
    call cu_sarg_ptr (argno, aptr, alen, ec);
    if ec = 0 then do;
        put skip list ("Specify value for clos_period");
        put skip (2);
        return;
    end;
else if index (arg, "--") = 0 then do;
    put skip list ("Specify value for clos_period.");
    put skip (2);
    return;
end;
else clos_period = float (arg);
topstring (3) = "Using Closure Period "||arg;
end;
else if arg = "-pm" then do;
    price_mult = true;
    argno = argno + 1;
    call cu_sarg_ptr (argno, aptr, alen, ec);
    if ec = 0 then do;
        put skip list ("A multiplier must be specified for -price_mult.");
        put skip (2);
        return;
    end;
    price_multiplier = float (arg);
end;
else do;
    put skip list ("Args are: -ies -nost -78 -offs -ons -cutoff <amt> -pe <amt> -ww -cp <n>");
    put skip (2);
    return;
end;
argno = argno + 1;
go to argip;

allocate pstruc, wstruc, nontech_struc;
call read_data;

nontech_struc.wsw = wsw;
nontech_struc.swg_mode = swgsw;
do smt = 1 to nme_smt;
    ind_struct (smt).ind_name = pstruc (smt).industry;
    temp, tprod = 0;
    do i = 1 to 2;
        do j = 1 to nmodels;
            tprod = tprod + pstruc (smt).pdata (i, j).prod * pstruc (smt).pdata (i, j).plants;
            temp = temp + pstruc (smt).pdata (i, j).employment * pstruc (smt).pdata (i, j).plants;
        end;
    end;
    do nstat = 1 to 2;
        do i = 1 to nmodels;
            if tprod = 0 then ind_struct (smt).value_add (i) = ind_struct (smt).value_add (i) +
                pstruc (smt).value_add + pstruc (smt).pdata (nstat, i).plants / tprod;
            else if temp = 0 then ind_struct (smt).value_add (i) = ind_struct (smt).value_add (i) +
                pstruc (smt).value_add + pstruc (smt).pdata (nstat, i).plants / temp;
        end;
    end;
end;

```

Xpage:
start:

```

end;
do k = 1 to 2;
  do m = 1 to wstruc (smt).sl (k).num_groups;
    do i = 1 to 5;
      test (k, m).mtcode (i).cd = "";
      code_grp (i) = 0;
      test (k, m).mtcode (i).nwsa = 0;
    end;
    do i = 1 to pstruc (smt).nwastes; /* decode waste code info */
      if wstruc (smt).sl (k).group (m).tech (i) = 0 then
        do j = 1 to 5;
          if pstruc (smt).waste (i).mtcode = codes (j) then code_grp (j) = code_grp (j) + 1;
        end;
      end;
      test (k, m).number = 0;
      do l = 1 to 5;
        if code_grp (l) > 0 then do;
          test (k, m).number = test (k, m).number + 1;
          test (k, m).mtcode (test (k, m).number).cd = codes (l);
          test (k, m).mtcode (test (k, m).number).nwsa =
            test (k, m).mtcode (test (k, m).number).nwsa + code_grp (l);
        end;
      end;
    end;
  end;
end;

nontech_struct.num_gen = 1;
do nstat = 1 to 2;
  call setup_wload;
  do l = 1 to nmodels;
    if pstruc (smt).pdata (nstat, l).plants = 0 then go to new_model;
    onsite (*, *, *, *) = "0-b";
    costs (*, *, *, *) = 0;
    do k = 1 to 2;
      do m = 1 to wstruc (smt).sl (k).num_groups;
        ncosts (*, *, *, *) = 0e0;
        nontech_struct.num_codes = test (k, m).number;
        do n = 1 to nontech_struct.num_codes;
          nontech_struct.test (n).code = test (k, m).mtcode (n).cd;
          nontech_struct.test (n).nwsa = test (k, m).mtcode (n).nwsa;
        end;
        non_tech_array (*, *, *) = 0e0;
        call nontech_iss (nt_ptr, addr (non_tech_array));
        if l < 4 & sum (nontech_struct.test (*).nwsa) = 0 then
          non_tech_array (4, ry, nfr) = non_tech_array (4, ry, nfr) + nontech_struct.num_gen;
        num_disp_types = allocate_waste (k, m);
        do n = 1 to num_disp_types;
          dp = prac_struct (n).disp; /* get current disp code */
          if dp = 0 then do;
            if -(dp = 6 & noslew) then do;
              size_var (n) = prac_struct (n).surf_area;
              call get_size (dp, (size_var (n)), sav (n));
            end;
          else do;
            call get_size (dp, prac_struct (n).tot_waste, sav (n));
            if prac_struct (n).num_gen_ons = 0 then size_var (n) = 0;
            else size_var (n) = prac_struct (n).tot_waste_ons;
          end;
        end;
      end;
    end;
  end;
end;

```

• 500:


```

end;
if prac_struct(n).num_gen_ons = 0 then do;
  ssv(n) = maxsize(dp);
end;
nontech_struct.num_gen_ons = prac_struct(n).num_gen_ons;
nontech_struct.num_gen_offs = prac_struct(n).num_gen_offs;
nontech_struct.tw = prac_struct(n).tot_waste;
nontech_struct.two = prac_struct(n).tot_waste_ons;
nontech_struct.tw_offs = prac_struct(n).tot_waste_offs;
nontech_struct.num_wastes = prac_struct(n).num_wastes;
nontech_struct.disp_code = dp;
nontech_struct.size_var = ssv(n);
if nontech_struct.num_gen_offs > 0 then
  nontech_struct.avgw2 = nontech_struct.tw_offs /
else nontech_struct.avgw2 = 0;
if lss then do;
  if i < 4 then do;
    call nontech_lss_incremental(nt_ptr, addr(nta));
    do x = 1 to 5;
      do y = 1 to 4;
        do z = 1 to 8;
          ncosts(n, x, y, z) = nta(x, y, z);
        end; /* end z */
      end; /* end y */
    end; /* end x */
    if whr(n) = 1 then do;
      if dp = 1 | dp = 5 | /* land-fill */
      dp = 8 | dp = 9 | dp = 10 then do;
        non_tech_array(4, ry, tr) = non_tech_array(4, ry,
          runoff_(dp, size_var(n)) * prac_struct(n).num_0
          non_tech_array(4, ry, tr) = non_tech_array(4, ry,
          closure_(dp, size_var(n)) * prac_struct(n).nu
          non_tech_array(4, ry, cp) = non_tech_array(4, ry,
          (fence_(dp, size_var(n)) * prac_struct(n).nu
        end;
      else if dp = 6 then do; /* disposal impoundments */
        non_tech_array(4, ci, nfr) = non_tech_array(4, ci,
          pstruc(smt).sla(nstat, i).num_disp_pp;
        non_tech_array(4, cap, tr) = non_tech_array(4, cap
          runoff_(6, size_var(n)) *
          pstruc(smt).sla(nstat, i).num_disp_pp;
        non_tech_array(4, cap, cp) = non_tech_array(4, cap
          fence_(6, size_var(n)) *
          max(i, pstruc(smt).sla(nstat, i).num_disp_pp
        end; /* end disposal group */
      else do; /* all other methods */
        if dp > 0 then do;
          non_tech_array(4, cap, cp) = non_tech_array(4

```



```

    * cap. cp) +
    n).num_gen_ons;

    sv (4, cap, tr) +
    num_gen_ons;

    (fence_ (dp, size_var (n))) * prec_struct (
    if dp ^= 32dp ^= 4 then /* not incin's */
    non_tech_array (4, cap, tr) = non_tech_arr
    runoff_ (dp, size_var (n))*prec_struct (n).

    end; /* end all other but rec */
    end; /* end onsite */
    else do;
    if wsw then do;
    call get_size_ (dp, nontech_struct.avgw2, tsz);
    tsz = min (4, tsz);
    annual (n) = tank_closure (tsz) * 2 + 1e2;
    end;
    non_tech_array (4, ry, tr) = non_tech_array (4, ry, tr) +
    iss_offs (dp) * price_multiplier
    * prec_struct (n).tot_waste_offs;
    end; /* end offs */
    end; /* end non swg do */
    else do; /* all costs for SWG's (waste analysis & treatment tanks) */
    call nontech_iss_swg (nt_ptr, addr (nta));
    do x = 1 to 5;
    do y = 1 to 4;
    do z = 1 to 8;
    non_tech_array (x, y, z) = non_tech_array (x, y
    nta (x, y, z);
    end;
    end;
    end;
    non_tech_array (4, ry, tr) = non_tech_array (4, ry, tr) +
    iss_offs (dp) * price_multiplier *
    prec_struct (n).tot_waste_offs;
    end; /* end swg */
    /* end iss */
    /* end nosi do */
    /* end no recycle */
    /* end n */

    end;
    if 1 < 4 then do;
    bf = left;
    onbit = '0'b;
    offbit = '0'b;
    p = 0; r = 0; w = 0; t = 0;
    do n = 1 to num_disp_types;
    if prec_struct (n).disp ^= 0 then do;
    if whr (n) = 1 then do;
    w = n;
    if sw (n) < 5 then r = n;
    onbit = '1'b;
    end;
    else do;
    offbit = '1'b;
    t = n;
    if wsw then do;
    if annual (n) < bf then do;
    p = n;

```

```

        bf = annual (n);
    end;
end;

end;
end;

if r = 0 then r = w;
if p = 0 then p = t;
do n = r, p;
    if (n = r & onsb1) | (n = p & offsb1) then do;
        nontech_struct.num_gen_ons = prac_struct (n).num_gen_ons;
        nontech_struct.num_gen_offs = prac_struct (n).num_gen_offs;
        nontech_struct.tw = prac_struct (n).tot_waste;
        nontech_struct.two = prac_struct (n).tot_waste_ons;
        nontech_struct.tw_offs = prac_struct (n).tot_waste_offs;
        nontech_struct.num_wastes = prac_struct (n).num_wastes;
        nontech_struct.disp_code = prac_struct (n).disp;
        nontech_struct.size_code = ssv (n);
        nontech_struct.size_var = size_var (n);
        if n = r then call nontech_iss_sonsite (nt_ptr, addr (nta));
        else call nontech_iss_offsite (nt_ptr, addr (nta));
        do x = 1 to 5;
            do y = 1 to 4;
                do z = 1 to 8;
                    ncosts (n, x, y, z) = nta (x, y, z);
                end;
            end;
        end;
    end;
end;

/* end do r, p */
/* end not swg */

end;
end;

do x = 1 to 5;
    do y = 1 to 4;
        do z = 1 to 8;
            costs (k, x, y, z) = costs (k, x, y, z) +
                (sum (ncosts (% x, y, z)) +
                nontech_array (x, y, z)) * wstruct (smt).sl (k).group (m).pct;
            /* end z */
            /* end y */
            /* end x */
            /* end m */
            /* end k */
        end;
    end;
end;

end;
end;

if noslew then do;
    ts_plus_both = pstruct (smt).sia (nstat, 1).both_imp + pstruct (smt).sia (nstat, 1).ts_imp;
    if ts_plus_both = 0 & i < 4 then do;
        sa_per_ts = pstruct (smt).sia (nstat, 1).ts_sa/pstruct (smt).sia (nstat, 1).num_ts_pp;
        num_disp_types = i;
        prac_struct (num_disp_types).disp = 7;
        prac_struct (num_disp_types).surf_area = sa_per_ts;
        size_var (7) = sa_per_ts;
        nontech_struct.num_gen_ons = 1;
        nontech_struct.num_gen_offs = 0;
        nontech_struct.tw = 1;
        nontech_struct.two = 1;
        nontech_struct.tw_offs = 0;
        nontech_struct.num_wastes = 1;
        nontech_struct.disp_code = 7;
        nontech_struct.size_var = sa_per_ts;
        call get_size_ (nontech_struct.disp_code, (size_var (7)), nontech_struct.size_code);
        call nontech_iss_incremental (nt_ptr, addr (nt2));
    end;
end;

```

```

nt2 (4, cl, nfr) = nt2 (4, cl, nfr) * pstruc (smt).sla (nstat, 1).num_ts_pp;
nt2 (4, cap, tr) = nt2 (4, cap, tr) +
runoff_ (7, se_per_ts) * pstruc (smt).sla (nstat, 1).num_ts_pp;
nt2 (4, cap, cp) = nt2 (4, cap, cp) +
fence_ (7, se_per_ts) * max (1, pstruc (smt).sla (nstat, 1).num_ts_pp * .5);

end;
else nt2 = 0;
/* end nosl do */

end;
else nt2 = 0;
do nsec = 1 to 5;
do frc = 1 to 4;
do act = 1 to 8;
carrys$ (smt, nstat, 1, nsec, frc, act) =
pstruc (smt).sla (nstat, 1).no_imp * costs (1, nsec, frc, act) +
pstruc (smt).sla (nstat, 1).disp_imp * costs (2, nsec, frc, act) +
pstruc (smt).sla (nstat, 1).ts_imp * (nt2 (nsec, frc, act) +
costs (1, nsec, frc, act)) +
pstruc (smt).sla (nstat, 1).both_imp * (costs (2, nsec, frc, act) +
nt2 (nsec, frc, act));

t1 =
pstruc (smt).sla (nstat, 1).no_imp * costs (1, nsec, frc, act);
t2 =
pstruc (smt).sla (nstat, 1).disp_imp * costs (2, nsec, frc, act) +
pstruc (smt).sla (nstat, 1).ts_imp * (nt2 (nsec, frc, act) +
costs (1, nsec, frc, act)) +
pstruc (smt).sla (nstat, 1).both_imp * (costs (2, nsec, frc, act) +
nt2 (nsec, frc, act));

if nstat = 1 then do; /* filter t1, t2 */
t1 = t1 * state_regs (nsec, act);
t2 = t2 * state_regs (nsec, act);
end;

slicost$ (smt, 1, nstat, 1, frc) = slicost$ (smt, 1, nstat, 1, frc) + t1;
slicost$ (smt, 2, nstat, 1, frc) = slicost$ (smt, 2, nstat, 1, frc) + t2;
/* end nsec, frc, act */
/* no sl, sl */

end; end; end;

do m = 1 to wstruc (smt).sl (k).num_groups;
sum_wload = sum (wload (nstat, 1, *));
do j = 1 to pstruc (smt).nwastes;
dp = wstruc (smt).sl (k).group (m).tech (j); /* get disp code */
if dp = 1; dp = 5 then sc = 3; /* sanitary */
else if dp = 8; dp = 10 then sc = 2; /* dump */
else if dp = 9 then sc = 2; /* secure */
else if dp = 2 then sc = 5; /* landfill */
else if dp = 3 then sc = 6; /* chlor. incinerator */
else if dp = 4 then sc = 7; /* hydrocarbon incin */
else if dp = 6 then sc = 8; /* surf. imp */
else if dp = 0 then sc = 1; /* recycle */
else sc = 0; /* mystery */
if sc = 0 then do;
if sum_wload = 0 then
gen = pstruc (smt).pdata (nstat, 1).plants *
wload (nstat, 1, j) / sum_wload;
wstruc (smt).sl (k).group (m).orig_pct;
else gen = 0;
waste = gen * sum_wload;
if onsite (k, j, m) then do;
stat_block$ (smt).size (1).gen_ons (sc) =
stat_block$ (smt).size (1).gen_ons (sc) + gen;
stat_block$ (smt).size (1).waste_ons (sc) =
stat_block$ (smt).size (1).waste_ons (sc) + waste;
end;
end;
end;
end;

```

```

end;
else do;
    stat_block$ (sat).size (i).gen_offs (sc) =
        stat_block$ (sat).size (i).gen_offs (sc) + gen;
    stat_block$ (sat).size (i).waste_offs (sc) =
        stat_block$ (sat).size (i).waste_offs (sc) + waste;
    /* end do */
end;
/* end sc group */
/* end j loop */
/* end m loop */
/* end k */
/* end i, nstat */

/* end of segment loop */

new_model:    end; end;
new:          end;

/* re-allocation and addition */

free pstruc, nontech_struc;

%page;
read_data: procedure;

dcl (nplants, ind, loop, sl_present, i, j, k, l, tech (14)) fixed bin;
    (np, pr, emp, vadd, pct1, vol, pct, percent (2, 7) init ((14)0)) float dec (6);
    ind_name char (60) var, mname char (3) var, superpercent float dec;

/* program to read data for ADL industries and place in pstruc global structure */

    on endfile (datafile) begin;
        close file (datafile);
        close file (datafile2);
        go to done;
    end;

    loop = 0;
    open file (datafile) title ("vfile_adi.data");
    open file (datafile2) title ("vfile_sta.data");

    do while (true);
        get file (datafile) list (ind_name, vadd, nplants, nup, nupr);
        loop = loop + 1;
        pstruc (loop).industry = ind_name;
        pstruc (loop).value_add = vadd;
        do i = 1 to 2;
            do j = 1 to 4;
                pstruc (loop).pdata (i, j).plants = 0;
                pstruc (loop).pdata (i, j).employment = 0;
                pstruc (loop).pdata (i, j).prod = 0;
                pstruc (loop).pdata (i, j).vol_waste = 0;
                pstruc (loop).pdata (i, j).level_one = 0;
            end;
        end;
        do j = 1 to 2;
            do i = 1 to nplants;
                get file (datafile) list (mname, np, pr, emp, vol, pct1);
                if mname = "L" then ind = 1;
                else if mname = "M" then ind = 2;
                else if mname = "S" then ind = 3;
                else ind = 4;
                pstruc (loop).pdata (j, ind).plants = np;
            end;
        end;
    end;

```

```

    pstruc (loop).pdata (j, ind).prod = pri;
    pstruc (loop).pdata (j, ind).employment = emp;
    pstruc (loop).pdata (j, ind).vol_waste = vol;
    pstruc (loop).pdata (j, ind).level_one = pct1;
    end;
    /* end plant loop */
    /* end reg/nonreg loop */
end;

do i = 1 to 3;
    get file (datafile) list (vol, emp, np);
    pstruc (loop).cutoff (i).waste = vol;
    pstruc (loop).cutoff (i).employment = emp;
    pstruc (loop).cutoff (i).plants = np;
end;

/* cut-off read loop */

pstruc (loop).waste (*).volume = 0e0;
pstruc (loop).nwastes = nws;
do i = 1 to nws;
    get file (datafile) list (pstruc (loop).waste (i).wname,
    (pstruc (loop).waste (i).volume,
    pstruc (loop).waste (i).cprc (j));
    (pstruc (loop).waste (i).cpct (j) do j = 1 to 3);
    pstruc (loop).waste (i).new_prc (j);
    pstruc (loop).waste (i).new_pct (j) do j = 1 to 2);
    pstruc (loop).waste (i).mtcode;
    pstruc (loop).waste (i).pct_ons);
end;

/* end waste loop */

/* read waste groups section of data and insert in wstruc structure */

wstruc (loop).sl (*).num_groups = 0;
percent (*, *) = 0;
do i = 1 to ngr;
    tech (*) = 0;
    get file (datafile) list
    (pct, tech (j) do j = 1 to nws));
    k = 1;
    do j = 1 to nws while (k = 1);
        if tech (j) = 6 then k = 2;
    end;
    wstruc (loop).sl (k).num_groups =
    wstruc (loop).sl (k).num_groups + 1;
    i = wstruc (loop).sl (k).num_groups;
    percent (k, i) = pct;
    wstruc (loop).sl (k).group (i).orig_pct = pct;
    do j = 1 to nws;
        wstruc (loop).sl (k).group (i).tech (j) = tech (j);
    end;
    /* end i loop */
    /* end j loop */
    /* calculate percent as normalized over all groups */
    sumpercent = sum (percent (k, *));
    do l = 1 to wstruc (loop).sl (k).num_groups;
        wstruc (loop).sl (k).group (l).pct =
        percent (k, l) / sumpercent;
    end;
    /* end l loop */
    /* end k loop */
end;

/* read surface impoundment data */

get file (datafile2) list (ind_name, vadd, sl_present);
do j = 1 to 4;

```



```

do i = 1 to 2;
  get file (datafile2) list (pstruc (loop).sia (i, j).no_inp,
    pstruc (loop).sia (i, j).disp_inp, pstruc (loop).sia (i, j).ts_inp,
    pstruc (loop).sia (i, j).both_inp, pstruc (loop).sia (i, j).num_disp_pp,
    pstruc (loop).sia (i, j).disp_sa, pstruc (loop).sia (i, j).num_ts_pp,
    pstruc (loop).sia (i, j).ts_sa);
  /* end i loop */
  /* end j loop */
  /* end read loop */
end;

done:
  num_smt = loop;
  return;

end read_data;

Xpage;
setup_wload: procedure;
/* procedure to set num_generators and waste for model sizes */
dcl (i, j) fixed bin;
sumload float dec (6);
pctload (14) float dec (6);

do i = 1 to nmodels;
  do j = 1 to 14;
    wload (nstat, i, j) = 0;
  end;
end;

sumload = sum (pstruc (smt).waste (*).volume);
do i = 1 to pstruc (smt).nwastes;
  pctload (i) = pstruc (smt).waste (i).volume / sumload;
end;
do i = 1 to nmodels;
  do j = 1 to pstruc (smt).nwastes;
    wload (nstat, i, j) = pstruc (smt).pdata (nstat, i).vol_waste
      * pctload (j);
  end;
end;
return;
end setup_wload;

Xpage;
allocate_waste: procedure (k, m) returns (fixed bin);
/* procedure to interpret pstruc and allocate waste streams by
disposal modes */
dcl (ndc, x, k, m, dspt) fixed bin;
dcl place (0:num_disp_codes) fixed bin;

dcl i gen_struc (0:num_disp_codes) aligned,
  2 disp_code fixed bin,
  2 num_wastes fixed bin,
  2 gen float dec (6),
  2 gen_ons float dec (6),
  2 gen_offs float dec (6),
  2 tot_waste float dec (6),
  2 tot_waste_ons float dec (6);

```

```

2 tot_waste_offs float dec (6),
2 num_sl float dec (6),
2 surf_area float dec (6);

do j = 0 to num_disp_codes;
  place (j) = 0;
  gen_struct (j).disp_code = 0;
  gen_struct (j).num_wastes = 0;
  gen_struct (j).gen,
    gen_struct (j).gen_ons,
    gen_struct (j).gen_offs,
    gen_struct (j).tot_waste,
    gen_struct (j).tot_waste_ons,
    gen_struct (j).tot_waste_offs,
    gen_struct (j).num_sl,
    gen_struct (j).surf_area = 0e0;
end;

do j = 1 to pstruc (smt).nwastes;
  dspt = wstruc (smt).sl (h).group (m).tech (j);
  gen_struct (dspt).disp_code = dspt;
  gen_struct (dspt).num_wastes = gen_struct (dspt).num_wastes + 1;
  gen_struct (dspt).gen = 1;
  gen_struct (dspt).tot_waste = gen_struct (dspt).tot_waste +
    wload (nstat, i, j);
  if dspt = 6 then do;
    di_plus_both = pstruc (smt).sla (nstat, i).both_imp + pstruc (smt).sla (nstat, i).disp_imp;
    if di_plus_both > 0 then do;
      if pstruc (smt).sla (nstat, i).num_disp_pp ~= 0 then
        gen_struct (dspt).surf_area =
          pstruc (smt).sla (nstat, i).disp_aa /
          pstruc (smt).sla (nstat, i).num_disp_pp;
      else gen_struct (dspt).surf_area = 0;
    end;
  end;
end;

end;
do j = 0 to num_disp_codes;
  if gen_struct (j).tot_waste >= breakeven (j) then do;
    gen_struct (j).tot_waste_ons = gen_struct (j).tot_waste;
    place (j) = 1;
    do x = 1 to pstruc (smt).nwastes;
      if wstruc (smt).sl (k).group (m).tech (x) =
        gen_struct (j).disp_code
        then onsite (k, x, m) = true;
    end;
  end;
  else do;
    place (j) = 2;
    gen_struct (j).tot_waste_offs = gen_struct (j).tot_waste;
    gen_struct (j).gen_offs = 1;
  end;
end;

end;
ndc = 0;
whr (+) = 0;
do j = 0 to num_disp_codes;
  if gen_struct (j).gen > 0 then do;
    ndc = ndc + 1;
    prac_struct (ndc) = gen_struct (j);
    whr (ndc) = place (j);
  end;
end;

```

```
and;  
end;  
return (ndc);  
end allocate_waste;
```

```
end rcra_cost;
```

```

/* ..... */
/* ..... */
/* Written 1980 at:  Arthur D. Little, Inc.
/*                  Cambridge, Ma. 02140
/* ..... */
/* ..... */

```

```
rcra_stat: procedure;
```

```
  Xinclude rcra;
```

```

dcl 1 sizes (4) aligned,
    2 gen_ons (nm) float dec(6),
    2 gen_offs (nm) float dec(6),
    2 waste_ons (nm) float dec(6),
    2 waste_offs (nm) float dec(6);
/* local structure */

```

```

dcl 1 i, j, n, argno init (1), alien fixed bin,
    nm init (8) static fixed bin,
    ec fixed bin (35);
dcl 1 t1 (nm, 4), t2 (nm, 4), t3 (nm, 4), t4 (nm, 4) float dec (6);
dcl 1 sizes (4) char (8) var init ("Large", "Medium", "Small", "SQ"),
    arg char (alien) unaligned based (aptr),
    (mdsw, hisw) bit (1) aligned init ("0-b"),
    aptr pointer;
dcl 1 hi_incr fixed bin init (0);

```

```

dcl 1 (sysprint output, report output, sysin input) file stream,
    (sum, trunc) builtin,
    endpage condition;

```

```

dcl cu_sarg_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35));
dcl header_ext entry (char (64) var, file);
%page;

```

```

on endpage (report) begin;
  put file (report) page;
  call header_ ("RCRA Generator & Waste Statistics", report);
  put file (report) edit ("Industry", "total")
    1col (1), a, col (128), a);
  put file (report) skip;
  put file (report) edit ("-----") (col (1), a)
    ("-----") (col (128), a);
  put file (report) skip;
  put file (report) edit ("
    hc incin", "surf imp", "
    dump", " san fill", " landfarm", "chl incin",
    (col (33), (nml)a (10));
  put file (report) skip (2);
end;

/* end begin */

```

arglp;

```

call cu_sarg_ptr (argno, aptr, alen, ec);
if ec = 0 then go to proceed;
else if arg = "-models" then mds = "1"b;
else if arg = "-hl" then hlsw = "1"b;
else do;
  put skip list ("Args are: -models -hl");
  put skip (2);
  return;
end;
argno = argno + 1;
go to arglp;

```

/* end error do */

/* initialization */

proceed:

```

open file (report) title ("vfile_ stat-report") print linesize (135) pagesize (50);
signal endpage (report);

```

```

if hlsw then hl_incr = num_hl_seg;
format (

```

```

col (30). (9) (x (3). p"zzz.zzz", x (4). p"zzz.zzz");
format (

```

```

col (30). (8) (p"zz.zzz.zzz", p"zzz.zzz.zzz");
do i = 1 to num_segments + hl_incr;
  do n = 1 to 4;
    do j = 1 to nr;
      lsize (n).gen_ons (j) = trunc (stat_block$ (i).size (n).gen_ons (j) + .5);
      lsize (n).gen_offs (j) = trunc (stat_block$ (i).size (n).gen_offs (j) + .5);
      lsize (n).waste_ons (j) = trunc (stat_block$ (i).size (n).waste_ons (j) + .5);
      lsize (n).waste_offs (j) = trunc (stat_block$ (i).size (n).waste_offs (j) + .5);
    end;
  end;
end;

```

```

end;
put file (report) list (ind_struc$ (i).ind_name);
put file (report) skip;
if md = 3 i <= num_segments then do n = 1 to 4;
  put file (report) edit (size (n)) ("Generator(ons):") (col (5), a)
  ((size (n).gen_ons (j)) do j = 1 to nm), sum (size (n).gen_ons (j)) (r (11));
  put file (report) edit (size (n)) ("Generator(off):") (col (5), a)
  ((size (n).gen_offs (j)) do j = 1 to nm), sum (size (n).gen_offs (j)) (r (11));
  sum (size (n).gen_ons (j)) + sum (size (n).gen_ons (j)) (col (12), p"zz.zzz.zzz");
  put file (report) skip;
  put file (report) edit ("Waste(ons):") (col (6), a)
  ((size (n).waste_ons (j)) do j = 1 to nm), sum (size (n).waste_ons (j)) (r (12));
  put skip file (report) edit ("Waste(off):") (col (6), a)
  ((size (n).waste_offs (j)) do j = 1 to nm), sum (size (n).waste_offs (j)) (r (12));
  sum (size (n).waste_offs (j)) + sum (size (n).waste_offs (j)) (col (12), p"zz.zzz.zzz");
  put file (report) skip;
end;

```

/* end n loop */

```

end;
put file (report) skip;
put file (report) edit ("Total Generators(ons):") (col (5), a)
((sum (size (n).gen_ons (j)) do j = 1 to nm), sum (size (n).gen_ons (j)) (r (11));
put skip file (report) edit ("Total Generators(off):") (col (5), a)
((sum (size (n).gen_offs (j)) do j = 1 to nm), sum (size (n).gen_offs (j)) (r (11));
sum (size (n).gen_offs (j)) + sum (size (n).gen_offs (j)) (col (12), p"zz.zzz.zzz");
put file (report) skip;
put file (report) edit ("Total Waste(ons):") (col (6), a)
((sum (size (n).waste_ons (j)) do j = 1 to nm), sum (size (n).waste_ons (j)) (r (12));

```



```

put skip file (report) edit ("Total Waste(off):") (col (6), a)
((sum (lsizes (*).waste_offs (j)) do j = 1 to nm). sum (lsizes (*).waste_offs (i))) (r (12))
((sum (lsizes (*).waste_offs (i)) sum (lsizes (*).waste_ons (*))) (col (122), p"zz.zzz.zz9");
put file (report) skip (2);
/* end i loop */
end;
do i = 1 to nm;
do j = 1 to 4;
t1 (i, j) = trunc (sum (stat_block$ (*).size (j).gen_ons (i)) + .5);
t2 (i, j) = trunc (sum (stat_block$ (*).size (j).gen_offs (i)) + .5);
t3 (i, j) = trunc (sum (stat_block$ (*).size (j).waste_ons (i)) + .5);
t4 (i, j) = trunc (sum (stat_block$ (*).size (j).waste_offs (i)) + .5);
end;
/* end j loop */
/* end i loop */
end;

put file (report) skip (1);
put file (report) list ("All Industries");
put file (report) skip;
if mdsu then do n = 1 to 4;
put file (report) edit (sizes (n)!!- Generator(ons):") (col (5), a)
((t1 (i, n)do i = 1 to nm). sum (t1 (*, n)) (r (11));
put skip file (report) edit (sizes (n)!!- Generator(off):") (col (5), a)
((t2 (i, n)do i = 1 to nm). sum (t2 (*, n)) (r (11))
((sum (t1 (*, n)) sum (t2 (*, n))) (col (122), p"zz.zzz.zz9");
put file (report) skip;
put file (report) edit ("Waste(ons):") (col (6), a)
((t3 (i, n)do i = 1 to nm). sum (t3 (*, n)) (r (12));
put file (report) edit ("Waste(off):") (col (6), a)
((t4 (i, n)do i = 1 to nm). sum (t4 (*, n)) (r (12))
((sum (t3 (*, n)) sum (t4 (*, n))) (col (122), p"zz.zzz.zz9");
put file (report) skip;
end;
/* end n */
put file (report) edit ("Total Generators(ons):") (col (5), a)
((sum (t1 (i, *)do i = 1 to nm). sum (t1 (*, *)) (r (11));
put skip file (report) edit ("Total Generators(off):") (col (5), a)
((sum (t2 (i, *)do i = 1 to nm). sum (t2 (*, *)) (r (11))
((sum (t1 (*, *)) sum (t2 (*, *)) (col (122), p"zz.zzz.zz9");
put file (report) skip;
put file (report) edit ("Total Waste(ons):") (col (6), a)
((sum (t3 (i, *)do i = 1 to nm). sum (t3 (*, *)) (r (12));
put skip file (report) edit ("Total Waste(off):") (col (6), a)
((sum (t4 (i, *)do i = 1 to nm). sum (t4 (*, *)) (r (12))
((sum (t3 (*, *)) sum (t4 (*, *)) (col (122), p"zz.zzz.zz9");
put file (report) skip;
close file (report);
end rcra_stat;

```


I-131

```

str3 char (100) var init ("-----");
arg char (alen) unaligned based (aptr),
(str2, old_ind, industry (2), ind_save) char (60) var,
filler char (3) var init (" "),
zone char (3) init (" "),
shift (4) char (8) var init ("AM", "PM", "NOON", "MIDNIGHT"),
mo (12) char (10) var init ("January", "February", "March", "April", "May", "June", "July",
"August", "September", "October", "November", "December"),
days (7) char (3) var init ("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun");
dcl (more_sat, repeat) bit (1) aligned init ("1'b");
dcl (industry_sw, model_sw) bit (1) aligned init ("0'b");
dcl aptr pointer;

dcl (substr, index, sum, length, char, clock, ltrim, lineno) builtin,
endpage condition,
(sysprint output, report output, datafile input) file stream,
cu_sar0_ptr ext entry (fixed bin, ptr, fixed bin, fixed bin (35)),
header_ext entry (char (64) var, file),
annualize_npv_ext entry (float dec, float dec, float dec, float dec) returns (float dec);
dcl decode_clock_value_sdate_time entry (fixed bin (71), fixed bin,
fixed bin, fixed bin, fixed bin, fixed bin, fixed bin (71),
fixed bin, char (3), fixed bin (35));
%page;
on endpage (report) begin;
line_num = lineno (report);
do while (line_num < 59);
put file (report) skip;
line_num = lineno (report);
end;
page_str = ltrim (char (pageno));
put file (report) edit (-"PAGE"- page_str) (col (125), a (4), x (1), a);
put file (report) page;
pageno = pageno + 1;
put file (report) edit (mo (month), dom, " ", year, hour, ":",
filler, ltrim (char (minute)), shift (sun), days (dow))
(col (109 - length (mo (month))), a, col (110), f (2), a (1), x (1), f (4),
x (1), f (2), a (1), a, a, x (1), a, x (1), a (3));
put file (report) skip;
call header_ ("ISS REPORT", report);
put file (report) skip (2);
put file (report) skip edit (str1) (col (65-length (str1)/2), a);
put file (report) skip edit (str3) (col (65-length (str3)/2), a);
/* begin block */

```

```

proceed:
open file (report) title ("vfile_sl_state.report") print linesize (135);
bits = clock ();
call decode_clock_value_sdate_time
(bits, month, dom, year, hour, minute, second, microsecond, dow, zone, code);
if hour = 12 then do;
if minute = 0 then sun = 3;
else sun = 2;
end;
else if hour = 0 then do;
hour = 12;
if minute = 0 then sun = 4;

```

```

end;
else if hour > 12 then do;
    hour = hour - 12;
    sun = 2;
end;
if minute < 10 then filler = "0";

smt = 0;
do while (more_smt);
    do n = 1 to 2;
        do mod = 1 to num_models;
            do nstat = 1 to 2;
                do si = 1 to 2;
                    do type = 1 to 2;
                        pctva (type, mod, nstat, si, n) = 0;
                        /* 1=first, 2=annual */
                        /* end type, si, nstat, mod, n */
                    end; end; end; end;
                    smt = smt + 2;
                    do n = 1 to 2 while (more_smt);
                        do mod = 1 to num_models;
                            do nstat = 1 to 2;
                                do si = 1 to 2;
                                    fye = sicost$ (smt+n-1, si, nstat, mod, 1) * 1e-6;
                                    ann = sicost$ (smt+n-1, si, nstat, mod, 2) * 1e-6;
                                    cap = sicost$ (smt+n-1, si, nstat, mod, 3) * 1e-6;
                                    clocst = sicost$ (smt+n-1, si, nstat, mod, 4) * 1e-6;
                                    fyc (mod, nstat, si, n) = fye + cap + clocst/clos_perlod + ann;
                                    annnc (mod, nstat, si, n) = annualize_npv_ (fye, ann, cap, clocst);
                                    if value_add (smt+n-1, mod, si, nstat) = 0 then do;
                                        pctva (1, mod, nstat, si, n) = fyc (mod, nstat, si, n) / value_add (smt+n-1, mod,
                                        si, nstat)*1e2;
                                        si, nstat)*1e2;
                                        pctva (2, mod, nstat, si, n) = annnc (mod, nstat, si, n) / value_add (smt+n-1, mod,
                                        si, nstat)*1e2;
                                    end;
                                else do;
                                    pctva (1, mod, nstat, si, n) = 0;
                                    pctva (2, mod, nstat, si, n) = 0;
                                end;
                                if smt+n > num_segments then more_smt = "0";
                                /* end si */
                                /* end nstat */
                                /* end mod */
                                /* end n */
                            end; end; end;
                            signal endpage (report);
                            do n = 1 to 2;
                                if smt+n-1 > num_segments then do;
                                    segment (n) = "";
                                    industry (n) = "";
                                end;
                                else do;
                                    segment (n) = substr (ind_struct$ (smt+n-1).ind_name,
                                    index (ind_struct$ (smt+n-1).ind_name, ",") + 1);
                                    industry (n) = substr (ind_struct$ (smt+n-1).ind_name,
                                    index (ind_struct$ (smt+n-1).ind_name, ",") - 1);
                                end;
                                put file (report) skip edit
                                (industry (1)) (col (48 - length (industry (1))/2), a)
                                (industry (2)) (col (103 - length (industry (2))/2), a)
                                (segment (1)) (skip, col (48 - length (segment (1))/2), a)
                                (segment (2)) (col (103 - length (segment (2))/2), a);
                            end;
                        end;
                    end;
                end;
            end;
        end;
    end;
end while (more_smt);

```



```

put file (report) skip;
put file (report) skip edit
  ("Plants with") (col (26), a)
  ("Plants without") (col (52), a)
  ("Plants with") (col (85), a)
  ("Plants without") (col (107), a);
do i = 26, 52, 85, 107;
  put file (report) edit ("Surface") (col (i), a);
end;
put file (report) skip edit ("Model") (a);
do i = 26, 52, 85, 107;
  put file (report) edit ("Impoundments") (col (i), a);
end;
put file (report) skip edit
  ("-----") (a)
  ("-----") (col (26), a)
  ("-----") (col (52), a)
  ("-----") (col (85), a)
  ("-----") (col (107), a);
do mod = 1 to num_models;
  put file (report) skip edit (sizestr (mod)) (a);
  put file (report) skip;
  do nstat = 1 to 2;
    put file (report) skip edit (reg (nstat)) (col (3), a);
    put file (report) skip;
    do type = 1 to 2;
      put file (report) skip edit
        (typestr (type)) (col (3), a)
        ("Cost as a % of") (skip, col (5), a)
        ("Value Added") (skip, col (5), a)
        (pctva (type, mod, nstat, 2, 1)) (col (30), p"zz9v.99")
        (pctva (type, mod, nstat, 1, 1)) (col (57), p"zz9v.99")
        (pctva (type, mod, nstat, 2, 2)) (col (90), p"zz9v.99")
        (pctva (type, mod, nstat, 1, 2)) (col (112), p"zz9v.99");
    end;
  end;
end;
end;
close file (report);
end sl_state;

```

tsdf_report.pl1

06/12/80 0948.9r w 06/04/80 1850.2

7632

```
tsdf_report: procedure;  
Xinclude rcra;  
dcl sysprint file external;  
  
dcl (rev_per_tsdf, num_tsdf) float dec;  
dcl (total_cost, total_waste, cost_to_waste) float dec;  
dcl (sum, trunc) builtin;  
  
total_cost = sum (carray$ (*, *, *, 4, 2, 1));  
total_waste = sum (stat_block$.size (*).waste_offs (*));  
  
cost_to_waste = total_cost / total_waste;  
num_tsdf = total_waste / 5e4;  
rev_per_tsdf = total_cost / num_tsdf;  
put skip edit  
  ("Total cost: ", total_cost) (skip, a, col (25), p"$$$,$$$,$$$"  
  ("Total waste: ", total_waste) (skip, a, col (25), p"zzz,zzz,zzz"  
  ("Number of TSDF's: ", trunc (num_tsdf)) (skip, a, col (25), p"zzz,zzz"  
  ("Cost to waste ratio: ", cost_to_waste) (skip, a, col (25), p"zzz,zzz"  
  ("Revenue per TSDF: ", rev_per_tsdf) (skip, a, col (25), p"$$$,$$$,$$$");  
put skip;  
end tsdf_report;
```